



# Open Source Used In Webex Teams Android Client 43.6

## **Cisco Systems, Inc.**

[www.cisco.com](http://www.cisco.com)

Cisco has more than 200 offices worldwide.  
Addresses, phone numbers, and fax numbers  
are listed on the Cisco website at  
[www.cisco.com/go/offices](http://www.cisco.com/go/offices).

Text Part Number: 78EE117C99-1682299314

**This document contains licenses and notices for open source software used in this product. With respect to the free/open source software listed in this document, if you have any questions or wish to receive a copy of any source code to which you may be entitled under the applicable free/open source license(s) (such as the GNU Lesser/General Public License), please submit this [form](#).**

**In your requests please include the following reference number 78EE117C99-1682299314**

## Contents

### **1.1 opus 1.3.1**

1.1.1 Available under license

### **1.2 boost 1.65.1**

1.2.1 Available under license

### **1.3 json-c 0.14**

1.3.1 Available under license

### **1.4 re2 2022-06-01**

1.4.1 Available under license

### **1.5 curl 7.88.1**

1.5.1 Available under license

### **1.6 ldns 1.13.1**

1.6.1 Available under license

### **1.7 okio 3.0.0**

1.7.1 Available under license

### **1.8 tidy 5.7.24**

1.8.1 Available under license

### **1.9 unbound 1.13.1**

1.9.1 Available under license

### **1.10 sqlite 3.40.1**

1.10.1 Available under license

### **1.11 openssl 1.1.1k**

1.11.1 Notifications

1.11.2 Available under license

### **1.12 libcxxabi 9.0.9svn**

1.12.1 Available under license

### **1.13 crashlytics 2.5.2**

- 1.13.1 Available under license
- 1.14 libilbc 2.0.2**
  - 1.14.1 Available under license
- 1.15 zlib 1.2.12**
  - 1.15.1 Available under license
- 1.16 appcompat 1.6.0**
  - 1.16.1 Available under license
- 1.17 openh264 3.35.0**
  - 1.17.1 Available under license
- 1.18 libcxx 9.0.9svn**
  - 1.18.1 Available under license
- 1.19 lifecycle-extensions 2.5.1**
  - 1.19.1 Available under license
- 1.20 guava 30.1.1-android**
  - 1.20.1 Available under license
- 1.21 junit 4.13.2**
  - 1.21.1 Available under license
- 1.22 google-play-services 18.0.1**
  - 1.22.1 Available under license
- 1.23 libxml2 2.9.10**
  - 1.23.1 Available under license
- 1.24 websocketpp 0.8.0**
  - 1.24.1 Available under license
- 1.25 safestring 2.0**
  - 1.25.1 Available under license
- 1.26 kotlin 1.6.10**
  - 1.26.1 Available under license
- 1.27 openssl 1.1.1t**
  - 1.27.1 Available under license
- 1.28 util-linux 2.31.1**
  - 1.28.1 Available under license
- 1.29 jansson 2.10**
  - 1.29.1 Available under license
- 1.30 minizip 1.01**
  - 1.30.1 Available under license
- 1.31 bouncy-castle 1.65**
  - 1.31.1 Available under license
- 1.32 okhttp 4.10.0**
  - 1.32.1 Available under license

### **1.33 protobuf-java 3.20.1**

1.33.1 Available under license

### **1.34 json-cpp 0.7.0**

1.34.1 Available under license

### **1.35 libsrtp 2.0.0**

1.35.1 Available under license

### **1.36 gson 2.8.7**

1.36.1 Available under license

### **1.37 firebase-messaging 31.4.0**

1.37.1 Available under license

### **1.38 jsoup 1.14.2**

1.38.1 Available under license

### **1.39 constraintlayout 2.1.4**

1.39.1 Available under license

## **1.1 opus 1.3.1**

### **1.1.1 Available under license :**

Copyright 2001-2011 Xiph.Org, Skype Limited, Octasic,  
Jean-Marc Valin, Timothy B. Terriberry,  
CSIRO, Gregory Maxwell, Mark Borgerding,  
Erik de Castro Lopo

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Internet Society, IETF or IETF Trust, nor the names of specific contributors, may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,

EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Opus is subject to the royalty-free patent licenses which are specified at:

Xiph.Org Foundation:

<https://datatracker.ietf.org/ipr/1524/>

Microsoft Corporation:

<https://datatracker.ietf.org/ipr/1914/>

Broadcom Corporation:

<https://datatracker.ietf.org/ipr/1526/>

## 1.2 boost 1.65.1

### 1.2.1 Available under license :

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 1.3 json-c 0.14

## 1.3.1 Available under license :

Copyright (c) 2009-2012 Eric Haszlakiewicz

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

-----  
Copyright (c) 2004, 2005 Metaparadigm Pte Ltd

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 1.4 re2 2022-06-01

## 1.4.1 Available under license :

```
// Copyright (c) 2009 The RE2 Authors. All rights reserved.
//
// Redistribution and use in source and binary forms, with or without
// modification, are permitted provided that the following conditions are
// met:
//
// * Redistributions of source code must retain the above copyright
// notice, this list of conditions and the following disclaimer.
// * Redistributions in binary form must reproduce the above
// copyright notice, this list of conditions and the following disclaimer
// in the documentation and/or other materials provided with the
// distribution.
// * Neither the name of Google Inc. nor the names of its
// contributors may be used to endorse or promote products derived from
// this software without specific prior written permission.
//
// THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
// "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
// LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
// A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
// OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
// SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
// LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
// DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
// THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
// (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
// OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
# This is the official list of people who can contribute
# (and typically have contributed) code to the RE2 repository.
# The AUTHORS file lists the copyright holders; this file
# lists people. For example, Google employees are listed here
# but not in AUTHORS, because Google holds the copyright.
#
# The submission process automatically checks to make sure
# that people submitting code are listed in this file (by email address).
#
# Names should be added to this file only after verifying that
# the individual or the individual's organization has agreed to
# the appropriate Contributor License Agreement, found here:
#
# http://code.google.com/legal/individual-cla-v1.0.html
# http://code.google.com/legal/corporate-cla-v1.0.html
#
# The agreement for individuals can be filled out on the web.
```

#  
# When adding J Random Contributor's name to this file,  
# either J's name or J's organization's name should be  
# added to the AUTHORS file, depending on whether the  
# individual or corporate CLA was used.

# Names should be added to this file like so:  
# Name <email address>

# Please keep the list sorted.

Dominic Battr <battre@chromium.org>  
Doug Kwan <dougkwan@google.com>  
Dmitriy Vyukov <dvyukov@google.com>  
John Millikin <jmillikin@gmail.com>  
Mike Nazarewicz <mpn@google.com>  
Nico Weber <thakis@chromium.org>  
Pawel Hajdan <phajdan.jr@gmail.com>  
Rob Pike <r@google.com>  
Russ Cox <rsc@swtch.com>  
Sanjay Ghemawat <sanjay@google.com>  
Stefano Rivera <stefano.rivera@gmail.com>  
Srinivasan Venkatachary <vsri@google.com>  
Viatcheslav Ostapenko <sl.ostapenko@samsung.com>

Apache License

Version 2.0, January 2004

<http://www.apache.org/licenses/>

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
- (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed

as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.

Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this

License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

## END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[ ]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

--- LLVM Exceptions to the Apache 2.0 License ----

As an exception, if, as a result of your compiling your source code, portions of this Software are embedded into an Object form of such source code, you may redistribute such embedded portions in such Object form without complying with the conditions of Sections 4(a), 4(b) and 4(d) of the License.

In addition, if you combine or link compiled forms of this Software with software that is licensed under the GPLv2 ("Combined Software") and if a court of competent jurisdiction determines that the patent provision (Section 3), the indemnity provision (Section 9) or other Section of the License conflicts with the conditions of the GPLv2, you may retroactively and prospectively choose to deem waived or otherwise exclude such Section(s) of

the License, but only in their entirety and only with respect to the Combined Software.

## 1.5 curl 7.88.1

### 1.5.1 Available under license :

#### COPYRIGHT AND PERMISSION NOTICE

Copyright (c) 1996 - 2023, Daniel Stenberg, <daniel@haxx.se>, and many contributors, see the THANKS file.

All rights reserved.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

## 1.6 Idns 1.13.1

### 1.6.1 Available under license :

Copyright (c) 2005,2006, NLnetLabs

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- \* Neither the name of NLnetLabs nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## 1.7 okio 3.0.0

### 1.7.1 Available under license :

No license file was found, but licenses were detected in source scan.

/\*

- \* Licensed to the Apache Software Foundation (ASF) under one or more
  - \* contributor license agreements. See the NOTICE file distributed with
  - \* this work for additional information regarding copyright ownership.
  - \* The ASF licenses this file to You under the Apache License, Version 2.0
  - \* (the "License"); you may not use this file except in compliance with
  - \* the License. You may obtain a copy of the License at
  - \*
  - \* <http://www.apache.org/licenses/LICENSE-2.0>
  - \*
  - \* Unless required by applicable law or agreed to in writing, software
  - \* distributed under the License is distributed on an "AS IS" BASIS,
  - \* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  - \* See the License for the specific language governing permissions and
  - \* limitations under the License.
- \*/

Found in path(s):

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/-Base64.kt

No license file was found, but licenses were detected in source scan.

/\*

- \* Copyright (C) 2020 Square, Inc. and others.
- \*
- \* Licensed under the Apache License, Version 2.0 (the "License");
- \* you may not use this file except in compliance with the License.
- \* You may obtain a copy of the License at
- \*
- \* <http://www.apache.org/licenses/LICENSE-2.0>

\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/CipherSource.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/CipherSink.kt  
No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2019 Square, Inc.  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.  
\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/Okio.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/BufferedSink.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/Source.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/BufferedSource.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/Buffer.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/RealBufferedSink.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/Sink.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/internal/Buffer.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/Timeout.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/RealBufferedSource.kt  
No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2017 Square, Inc.  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.  
\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/Utf8.kt  
No license file was found, but licenses were detected in source scan.

/\*  
\* Copyright (C) 2015 Square, Inc.  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.  
\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/SegmentedByteString.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/SegmentedByteString.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/ForwardingTimeout.kt  
No license file was found, but licenses were detected in source scan.

/\*  
\* Copyright (C) 2014 Square, Inc.  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.  
\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/Buffer.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/RealBufferedSource.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/GzipSource.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/InflaterSource.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/GzipSink.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/SegmentPool.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/ForwardingSink.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/BufferedSource.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/BufferedSink.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/Timeout.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/RealBufferedSink.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/SegmentPool.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/DeflaterSink.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/JvmOkio.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/AsyncTimeout.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/ForwardingSource.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/Segment.kt  
No license file was found, but licenses were detected in source scan.

/\*  
\* Copyright (C) 2019 Square, Inc.  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.  
\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

- \* See the License for the specific language governing permissions and
  - \* limitations under the License.
- \*/

Found in path(s):

- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/internal/RealBufferedSink.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/internal/SegmentedByteString.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/internal/RealBufferedSource.kt

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright 2014 Square Inc.

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

Found in path(s):

- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/ByteString.kt

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2018 Square, Inc.

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

Found in path(s):

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/-  
DeprecatedUtf8.kt

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/-  
DeprecatedUpgrade.kt

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-  
jar/commonMain/okio/PeekSource.kt

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/-  
DeprecatedOkio.kt

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/Throttler.kt

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2016 Square, Inc.

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

Found in path(s):

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/HashingSource.kt

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/Pipe.kt

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/Options.kt

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/HashingSink.kt

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2018 Square, Inc.

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

- \* See the License for the specific language governing permissions and
- \* limitations under the License.
- \*/

Found in path(s):

- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/internal/ByteString.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/-Util.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/-CommonPlatform.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/-JvmPlatform.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/ByteString.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/internal/Utf8.kt

No license file was found, but licenses were detected in source scan.

/\*

- \* Copyright (C) 2020 Square, Inc.
- \*
- \* Licensed under the Apache License, Version 2.0 (the "License");
- \* you may not use this file except in compliance with the License.
- \* You may obtain a copy of the License at
- \*
- \* <http://www.apache.org/licenses/LICENSE-2.0>
- \*
- \* Unless required by applicable law or agreed to in writing, software
- \* distributed under the License is distributed on an "AS IS" BASIS,
- \* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
- \* See the License for the specific language governing permissions and
- \* limitations under the License.
- \*/

Found in path(s):

- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/internal/Path.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/HashingSink.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/NioSystemFileSystem.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/Path.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/JvmSystemFileSystem.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/ExperimentalFileSystem.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/ForwardingFileSystem.kt
- \* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/HashingSource.kt

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/FileSystem.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/Path.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/FileMetadata.kt

No license file was found, but licenses were detected in source scan.

/\*

\* Licensed to the Apache Software Foundation (ASF) under one or more  
\* contributor license agreements. See the NOTICE file distributed with  
\* this work for additional information regarding copyright ownership.  
\* The ASF licenses this file to You under the Apache License, Version 2.0  
\* (the "License"); you may not use this file except in compliance with  
\* the License. You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/internal/ZipEntry.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/internal/zip.kt  
\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/ZipFileSystem.kt

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2021 Square, Inc.  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.  
\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1387809635\_1660200733.4071565/0/okio-3-0-0-sources-1-

```
jar/jvmMain/okio/internal/ResourceFileSystem.kt
* /opt/cola/permits/1387809635_1660200733.4071565/0/okio-3-0-0-sources-1-
jar/jvmMain/okio/internal/FixedLengthSource.kt
* /opt/cola/permits/1387809635_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/FileHandle.kt
* /opt/cola/permits/1387809635_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/JvmFileHandle.kt
* /opt/cola/permits/1387809635_1660200733.4071565/0/okio-3-0-0-sources-1-jar/commonMain/okio/internal/
FileSystem.kt
* /opt/cola/permits/1387809635_1660200733.4071565/0/okio-3-0-0-sources-1-jar/jvmMain/okio/FileSystem.kt
```

# 1.8 tidy 5.7.24

## 1.8.1 Available under license :

```
# HTML Tidy
```

```
## HTML parser and pretty printer
```

Copyright (c) 1998-2016 World Wide Web Consortium  
(Massachusetts Institute of Technology, European Research  
Consortium for Informatics and Mathematics, Keio University).  
All Rights Reserved.

Additional contributions (c) 2001-2016 University of Toronto, Terry Teague,  
@geoffmcl, HTACG, and others.

```
### Contributing Author(s):
```

```
Dave Raggett <dsr@w3.org>
```

The contributing author(s) would like to thank all those who  
helped with testing, bug fixes and suggestions for improvements.  
This wouldn't have been possible without your help.

```
## COPYRIGHT NOTICE:
```

This software and documentation is provided "as is," and  
the copyright holders and contributing author(s) make no  
representations or warranties, express or implied, including  
but not limited to, warranties of merchantability or fitness  
for any particular purpose or that the use of the software or  
documentation will not infringe any third party patents,  
copyrights, trademarks or other rights.

The copyright holders and contributing author(s) will not be held  
liable for any direct, indirect, special or consequential damages  
arising out of any use of the software or documentation, even if  
advised of the possibility of such damage.

Permission is hereby granted to use, copy, modify, and distribute this source code, or portions hereof, documentation and executables, for any purpose, without fee, subject to the following restrictions:

1. The origin of this source code must not be misrepresented.
2. Altered versions must be plainly marked as such and must not be misrepresented as being the original source.
3. This Copyright notice may not be removed or altered from any source or altered source distribution.

The copyright holders and contributing author(s) specifically permit, without fee, and encourage the use of this source code as a component for supporting the Hypertext Markup Language in commercial products. If you use this source code in a product, acknowledgement is not required but would be appreciated.

## 1.9 unbound 1.13.1

### 1.9.1 Available under license :

Copyright (c) 2009, Zdenek Vasicek (vasicek AT fit.vutbr.cz)  
Marek Vavrusa (xvavru00 AT stud.fit.vutbr.cz)

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- \* Neither the name of the organization nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) 2007, NLnet Labs. All rights reserved.

This software is open source.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the NLNET LABS nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) 2009, Dmitriy Demidov aka terminus. All rights reserved.

This software is open source.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the Dmitriy Demidov nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## 1.10 sqlite 3.40.1

### 1.10.1 Available under license :

No license file was found, but licenses were detected in source scan.

```
/*
** CAPI3REF: Configuration Options
** KEYWORDS: {configuration option}
**
** These constants are the available integer configuration options that
** can be passed as the first argument to the [sqlite3_config()] interface.
**
** New configuration options may be added in future releases of SQLite.
** Existing configuration options might be discontinued. Applications
** should check the return code from [sqlite3_config()] to make sure that
** the call worked. The [sqlite3_config()] interface will return a
** non-zero [error code] if a discontinued or unsupported configuration option
** is invoked.
**
** <dl>
** [[SQLITE_CONFIG_SINGLETHREAD]] <dt>SQLITE_CONFIG_SINGLETHREAD</dt>
** <dd>There are no arguments to this option. ^This option sets the
** [threading mode] to Single-thread. In other words, it disables
** all mutexing and puts SQLite into a mode where it can only be used
** by a single thread. ^If SQLite is compiled with
** the [SQLITE_THREADSAFE | SQLITE_THREADSAFE=0] compile-time option then
** it is not possible to change the [threading mode] from its default
** value of Single-thread and so [sqlite3_config()] will return
** [SQLITE_ERROR] if called with the SQLITE_CONFIG_SINGLETHREAD
** configuration option.</dd>
**
** [[SQLITE_CONFIG_MULTITHREAD]] <dt>SQLITE_CONFIG_MULTITHREAD</dt>
** <dd>There are no arguments to this option. ^This option sets the
** [threading mode] to Multi-thread. In other words, it disables
```

\*\* mutexing on [database connection] and [prepared statement] objects.  
 \*\* The application is responsible for serializing access to  
 \*\* [database connections] and [prepared statements]. But other mutexes  
 \*\* are enabled so that SQLite will be safe to use in a multi-threaded  
 \*\* environment as long as no two threads attempt to use the same  
 \*\* [database connection] at the same time. ^If SQLite is compiled with  
 \*\* the [SQLITE\_THREADSAFE | SQLITE\_THREADSAFE=0] compile-time option then  
 \*\* it is not possible to set the Multi-thread [threading mode] and  
 \*\* [sqlite3\_config()] will return [SQLITE\_ERROR] if called with the  
 \*\* SQLITE\_CONFIG\_MULTITHREAD configuration option.</dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_SERIALIZED]] <dt>SQLITE\_CONFIG\_SERIALIZED</dt>  
 \*\* <dd>There are no arguments to this option. ^This option sets the  
 \*\* [threading mode] to Serialized. In other words, this option enables  
 \*\* all mutexes including the recursive  
 \*\* mutexes on [database connection] and [prepared statement] objects.  
 \*\* In this mode (which is the default when SQLite is compiled with  
 \*\* [SQLITE\_THREADSAFE=1]) the SQLite library will itself serialize access  
 \*\* to [database connections] and [prepared statements] so that the  
 \*\* application is free to use the same [database connection] or the  
 \*\* same [prepared statement] in different threads at the same time.  
 \*\* ^If SQLite is compiled with  
 \*\* the [SQLITE\_THREADSAFE | SQLITE\_THREADSAFE=0] compile-time option then  
 \*\* it is not possible to set the Serialized [threading mode] and  
 \*\* [sqlite3\_config()] will return [SQLITE\_ERROR] if called with the  
 \*\* SQLITE\_CONFIG\_SERIALIZED configuration option.</dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_MALLOC]] <dt>SQLITE\_CONFIG\_MALLOC</dt>  
 \*\* <dd> ^The SQLITE\_CONFIG\_MALLOC option takes a single argument which is  
 \*\* a pointer to an instance of the [sqlite3\_mem\_methods] structure.  
 \*\* The argument specifies  
 \*\* alternative low-level memory allocation routines to be used in place of  
 \*\* the memory allocation routines built into SQLite.)^ ^SQLite makes  
 \*\* its own private copy of the content of the [sqlite3\_mem\_methods] structure  
 \*\* before the [sqlite3\_config()] call returns.</dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_GETMALLOC]] <dt>SQLITE\_CONFIG\_GETMALLOC</dt>  
 \*\* <dd> ^The SQLITE\_CONFIG\_GETMALLOC option takes a single argument which  
 \*\* is a pointer to an instance of the [sqlite3\_mem\_methods] structure.  
 \*\* The [sqlite3\_mem\_methods]  
 \*\* structure is filled with the currently defined memory allocation routines.)^  
 \*\* This option can be used to overload the default memory allocation  
 \*\* routines with a wrapper that simulates memory allocation failure or  
 \*\* tracks memory usage, for example. </dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_SMALL\_MALLOC]] <dt>SQLITE\_CONFIG\_SMALL\_MALLOC</dt>  
 \*\* <dd> ^The SQLITE\_CONFIG\_SMALL\_MALLOC option takes single argument of  
 \*\* type int, interpreted as a boolean, which if true provides a hint to

\*\* SQLite that it should avoid large memory allocations if possible.  
 \*\* SQLite will run faster if it is free to make large memory allocations,  
 \*\* but some application might prefer to run slower in exchange for  
 \*\* guarantees about memory fragmentation that are possible if large  
 \*\* allocations are avoided. This hint is normally off.  
 \*\* </dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_MEMSTATUS]] <dt>SQLITE\_CONFIG\_MEMSTATUS</dt>  
 \*\* <dd> ^The SQLITE\_CONFIG\_MEMSTATUS option takes single argument of type int,  
 \*\* interpreted as a boolean, which enables or disables the collection of  
 \*\* memory allocation statistics. ^(When memory allocation statistics are  
 \*\* disabled, the following SQLite interfaces become non-operational:  
 \*\* <ul>  
 \*\* <li> [sqlite3\_hard\_heap\_limit64()]  
 \*\* <li> [sqlite3\_memory\_used()]  
 \*\* <li> [sqlite3\_memory\_highwater()]  
 \*\* <li> [sqlite3\_soft\_heap\_limit64()]  
 \*\* <li> [sqlite3\_status64()]  
 \*\* </ul>)^  
 \*\* ^Memory allocation statistics are enabled by default unless SQLite is  
 \*\* compiled with [SQLITE\_DEFAULT\_MEMSTATUS]=0 in which case memory  
 \*\* allocation statistics are disabled by default.  
 \*\* </dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_SCRATCH]] <dt>SQLITE\_CONFIG\_SCRATCH</dt>  
 \*\* <dd> The SQLITE\_CONFIG\_SCRATCH option is no longer used.  
 \*\* </dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_PAGECACHE]] <dt>SQLITE\_CONFIG\_PAGECACHE</dt>  
 \*\* <dd> ^The SQLITE\_CONFIG\_PAGECACHE option specifies a memory pool  
 \*\* that SQLite can use for the database page cache with the default page  
 \*\* cache implementation.  
 \*\* This configuration option is a no-op if an application-defined page  
 \*\* cache implementation is loaded using the [SQLITE\_CONFIG\_PCACHE2].  
 \*\* ^There are three arguments to SQLITE\_CONFIG\_PAGECACHE: A pointer to  
 \*\* 8-byte aligned memory (pMem), the size of each page cache line (sz),  
 \*\* and the number of cache lines (N).  
 \*\* The sz argument should be the size of the largest database page  
 \*\* (a power of two between 512 and 65536) plus some extra bytes for each  
 \*\* page header. ^The number of extra bytes needed by the page header  
 \*\* can be determined using [SQLITE\_CONFIG\_PCACHE\_HDRSZ].  
 \*\* ^It is harmless, apart from the wasted memory,  
 \*\* for the sz parameter to be larger than necessary. The pMem  
 \*\* argument must be either a NULL pointer or a pointer to an 8-byte  
 \*\* aligned block of memory of at least sz\*N bytes, otherwise  
 \*\* subsequent behavior is undefined.  
 \*\* ^When pMem is not NULL, SQLite will strive to use the memory provided  
 \*\* to satisfy page cache needs, falling back to [sqlite3\_malloc()] if

\*\* a page cache line is larger than sz bytes or if all of the pMem buffer  
 \*\* is exhausted.

\*\* ^If pMem is NULL and N is non-zero, then each database connection  
 \*\* does an initial bulk allocation for page cache memory  
 \*\* from [sqlite3\_malloc()] sufficient for N cache lines if N is positive or  
 \*\* of -1024\*N bytes if N is negative, . ^If additional  
 \*\* page cache memory is needed beyond what is provided by the initial  
 \*\* allocation, then SQLite goes to [sqlite3\_malloc()] separately for each  
 \*\* additional cache line. </dd>

\*\*

\*\* [[SQLITE\_CONFIG\_HEAP]] <dt>SQLITE\_CONFIG\_HEAP</dt>

\*\* <dd> ^The SQLITE\_CONFIG\_HEAP option specifies a static memory buffer  
 \*\* that SQLite will use for all of its dynamic memory allocation needs  
 \*\* beyond those provided for by [SQLITE\_CONFIG\_PAGECACHE].  
 \*\* ^The SQLITE\_CONFIG\_HEAP option is only available if SQLite is compiled  
 \*\* with either [SQLITE\_ENABLE\_MEMSYS3] or [SQLITE\_ENABLE\_MEMSYS5] and returns  
 \*\* [SQLITE\_ERROR] if invoked otherwise.

\*\* ^There are three arguments to SQLITE\_CONFIG\_HEAP:

\*\* An 8-byte aligned pointer to the memory,  
 \*\* the number of bytes in the memory buffer, and the minimum allocation size.

\*\* ^If the first pointer (the memory pointer) is NULL, then SQLite reverts  
 \*\* to using its default memory allocator (the system malloc() implementation),  
 \*\* undoing any prior invocation of [SQLITE\_CONFIG\_MALLOC]. ^If the  
 \*\* memory pointer is not NULL then the alternative memory  
 \*\* allocator is engaged to handle all of SQLites memory allocation needs.

\*\* The first pointer (the memory pointer) must be aligned to an 8-byte  
 \*\* boundary or subsequent behavior of SQLite will be undefined.

\*\* The minimum allocation size is capped at 2\*\*12. Reasonable values  
 \*\* for the minimum allocation size are 2\*\*5 through 2\*\*8.</dd>

\*\*

\*\* [[SQLITE\_CONFIG\_MUTEX]] <dt>SQLITE\_CONFIG\_MUTEX</dt>

\*\* <dd> ^The SQLITE\_CONFIG\_MUTEX option takes a single argument which is a  
 \*\* pointer to an instance of the [sqlite3\_mutex\_methods] structure.

\*\* The argument specifies alternative low-level mutex routines to be used  
 \*\* in place the mutex routines built into SQLite.)^ ^SQLite makes a copy of  
 \*\* the content of the [sqlite3\_mutex\_methods] structure before the call to  
 \*\* [sqlite3\_config()] returns. ^If SQLite is compiled with  
 \*\* the [SQLITE\_THREADSAFE | SQLITE\_THREADSAFE=0] compile-time option then  
 \*\* the entire mutexing subsystem is omitted from the build and hence calls to  
 \*\* [sqlite3\_config()] with the SQLITE\_CONFIG\_MUTEX configuration option will  
 \*\* return [SQLITE\_ERROR].</dd>

\*\*

\*\* [[SQLITE\_CONFIG\_GETMUTEX]] <dt>SQLITE\_CONFIG\_GETMUTEX</dt>

\*\* <dd> ^The SQLITE\_CONFIG\_GETMUTEX option takes a single argument which  
 \*\* is a pointer to an instance of the [sqlite3\_mutex\_methods] structure. The  
 \*\* [sqlite3\_mutex\_methods]  
 \*\* structure is filled with the currently defined mutex routines.)^

\*\* This option can be used to overload the default mutex allocation

\*\* routines with a wrapper used to track mutex usage for performance  
 \*\* profiling or testing, for example. ^If SQLite is compiled with  
 \*\* the [SQLITE\_THREADSAFE | SQLITE\_THREADSAFE=0] compile-time option then  
 \*\* the entire mutexing subsystem is omitted from the build and hence calls to  
 \*\* [sqlite3\_config()] with the SQLITE\_CONFIG\_GETMUTEX configuration option will  
 \*\* return [SQLITE\_ERROR].</dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_LOOKASIDE]] <dt>SQLITE\_CONFIG\_LOOKASIDE</dt>  
 \*\* <dd> ^(The SQLITE\_CONFIG\_LOOKASIDE option takes two arguments that determine  
 \*\* the default size of lookaside memory on each [database connection].  
 \*\* The first argument is the  
 \*\* size of each lookaside buffer slot and the second is the number of  
 \*\* slots allocated to each database connection.)^ (SQLITE\_CONFIG\_LOOKASIDE  
 \*\* sets the <i>default</i> lookaside size. The [SQLITE\_DBCONFIG\_LOOKASIDE]  
 \*\* option to [sqlite3\_db\_config()] can be used to change the lookaside  
 \*\* configuration on individual connections.)^ </dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_PCACHE2]] <dt>SQLITE\_CONFIG\_PCACHE2</dt>  
 \*\* <dd> ^(The SQLITE\_CONFIG\_PCACHE2 option takes a single argument which is  
 \*\* a pointer to an [sqlite3\_pcache\_methods2] object. This object specifies  
 \*\* the interface to a custom page cache implementation.)^  
 \*\* ^SQLite makes a copy of the [sqlite3\_pcache\_methods2] object.</dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_GETPCACHE2]] <dt>SQLITE\_CONFIG\_GETPCACHE2</dt>  
 \*\* <dd> ^(The SQLITE\_CONFIG\_GETPCACHE2 option takes a single argument which  
 \*\* is a pointer to an [sqlite3\_pcache\_methods2] object. SQLite copies of  
 \*\* the current page cache implementation into that object.)^ </dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_LOG]] <dt>SQLITE\_CONFIG\_LOG</dt>  
 \*\* <dd> The SQLITE\_CONFIG\_LOG option is used to configure the SQLite  
 \*\* global [error log].  
 \*\* (^The SQLITE\_CONFIG\_LOG option takes two arguments: a pointer to a  
 \*\* function with a call signature of void(\*)(void\*,int,const char\*),  
 \*\* and a pointer to void. ^If the function pointer is not NULL, it is  
 \*\* invoked by [sqlite3\_log()] to process each logging event. ^If the  
 \*\* function pointer is NULL, the [sqlite3\_log()] interface becomes a no-op.  
 \*\* ^The void pointer that is the second argument to SQLITE\_CONFIG\_LOG is  
 \*\* passed through as the first parameter to the application-defined logger  
 \*\* function whenever that function is invoked. ^The second parameter to  
 \*\* the logger function is a copy of the first parameter to the corresponding  
 \*\* [sqlite3\_log()] call and is intended to be a [result code] or an  
 \*\* [extended result code]. ^The third parameter passed to the logger is  
 \*\* log message after formatting via [sqlite3\_snprintf()].  
 \*\* The SQLite logging interface is not reentrant; the logger function  
 \*\* supplied by the application must not invoke any SQLite interface.  
 \*\* In a multi-threaded application, the application-defined logger  
 \*\* function must be threadsafe. </dd>  
 \*\*

```

** [[SQLITE_CONFIG_URI]] <dt>SQLITE_CONFIG_URI
** <dd>^(The SQLITE_CONFIG_URI option takes a single argument of type int.
** If non-zero, then URI handling is globally enabled. If the parameter is zero,
** then URI handling is globally disabled.)^ ^If URI handling is globally
** enabled, all filenames passed to [sqlite3_open()], [sqlite3_open_v2()],
** [sqlite3_open16()] or
** specified as part of [ATTACH] commands are interpreted as URIs, regardless
** of whether or not the [SQLITE_OPEN_URI] flag is set when the database
** connection is opened. ^If it is globally disabled, filenames are
** only interpreted as URIs if the SQLITE_OPEN_URI flag is set when the
** database connection is opened. ^By default, URI handling is globally
** disabled. The default value may be changed by compiling with the
** [SQLITE_USE_URI] symbol defined.)^
**
** [[SQLITE_CONFIG_COVERING_INDEX_SCAN]] <dt>SQLITE_CONFIG_COVERING_INDEX_SCAN
** <dd>^The SQLITE_CONFIG_COVERING_INDEX_SCAN option takes a single integer
** argument which is interpreted as a boolean in order to enable or disable
** the use of covering indices for full table scans in the query optimizer.
** ^The default setting is determined
** by the [SQLITE_ALLOW_COVERING_INDEX_SCAN] compile-time option, or is "on"
** if that compile-time option is omitted.
** The ability to disable the use of covering indices for full table scans
** is because some incorrectly coded legacy applications might malfunction
** when the optimization is enabled. Providing the ability to
** disable the optimization allows the older, buggy application code to work
** without change even with newer versions of SQLite.
**
** [[SQLITE_CONFIG_PCACHE]] [[SQLITE_CONFIG_GETPCACHE]]
** <dt>SQLITE_CONFIG_PCACHE and SQLITE_CONFIG_GETPCACHE
** <dd> These options are obsolete and should not be used by new code.
** They are retained for backwards compatibility but are now no-ops.
** </dd>
**
** [[SQLITE_CONFIG_SQLLOG]]
** <dt>SQLITE_CONFIG_SQLLOG
** <dd>This option is only available if sqlite is compiled with the
** [SQLITE_ENABLE_SQLLOG] pre-processor macro defined. The first argument should
** be a pointer to a function of type void(*)(void*,sqlite3*,const char*, int).
** The second should be of type (void*). The callback is invoked by the library
** in three separate circumstances, identified by the value passed as the
** fourth parameter. If the fourth parameter is 0, then the database connection
** passed as the second argument has just been opened. The third argument
** points to a buffer containing the name of the main database file. If the
** fourth parameter is 1, then the SQL statement that the third parameter
** points to has just been executed. Or, if the fourth parameter is 2, then
** the connection being passed as the second parameter is being closed. The
** third parameter is passed NULL in this case. An example of using this
** configuration option can be seen in the "test_sqllog.c" source file in

```

\*\* the canonical SQLite source tree.</dd>

\*\*

\*\* [[SQLITE\_CONFIG\_MMAP\_SIZE]]

\*\* <dt>SQLITE\_CONFIG\_MMAP\_SIZE

\*\* <dd>^SQLITE\_CONFIG\_MMAP\_SIZE takes two 64-bit integer (sqlite3\_int64) values

\*\* that are the default mmap size limit (the default setting for

\*\* [PRAGMA mmap\_size]) and the maximum allowed mmap size limit.

\*\* ^The default setting can be overridden by each database connection using

\*\* either the [PRAGMA mmap\_size] command, or by using the

\*\* [SQLITE\_FCNTL\_MMAP\_SIZE] file control. ^(The maximum allowed mmap size

\*\* will be silently truncated if necessary so that it does not exceed the

\*\* compile-time maximum mmap size set by the

\*\* [SQLITE\_MAX\_MMAP\_SIZE] compile-time option.)^

\*\* ^If either argument to this option is negative, then that argument is

\*\* changed to its compile-time default.

\*\*

\*\* [[SQLITE\_CONFIG\_WIN32\_HEAPSIZE]]

\*\* <dt>SQLITE\_CONFIG\_WIN32\_HEAPSIZE

\*\* <dd>^The SQLITE\_CONFIG\_WIN32\_HEAPSIZE option is only available if SQLite is

\*\* compiled for Windows with the [SQLITE\_WIN32\_MALLOC] pre-processor macro

\*\* defined. ^SQLITE\_CONFIG\_WIN32\_HEAPSIZE takes a 32-bit unsigned integer value

\*\* that specifies the maximum size of the created heap.

\*\*

\*\* [[SQLITE\_CONFIG\_PCACHE\_HDRSZ]]

\*\* <dt>SQLITE\_CONFIG\_PCACHE\_HDRSZ

\*\* <dd>^The SQLITE\_CONFIG\_PCACHE\_HDRSZ option takes a single parameter which

\*\* is a pointer to an integer and writes into that integer the number of extra

\*\* bytes per page required for each page in [SQLITE\_CONFIG\_PAGECACHE].

\*\* The amount of extra space required can change depending on the compiler,

\*\* target platform, and SQLite version.

\*\*

\*\* [[SQLITE\_CONFIG\_PMASZ]]

\*\* <dt>SQLITE\_CONFIG\_PMASZ

\*\* <dd>^The SQLITE\_CONFIG\_PMASZ option takes a single parameter which

\*\* is an unsigned integer and sets the "Minimum PMA Size" for the multithreaded

\*\* sorter to that integer. The default minimum PMA Size is set by the

\*\* [SQLITE\_SORTER\_PMASZ] compile-time option. New threads are launched

\*\* to help with sort operations when multithreaded sorting

\*\* is enabled (using the [PRAGMA threads] command) and the amount of content

\*\* to be sorted exceeds the page size times the minimum of the

\*\* [PRAGMA cache\_size] setting and this value.

\*\*

\*\* [[SQLITE\_CONFIG\_STMTJRNL\_SPILL]]

\*\* <dt>SQLITE\_CONFIG\_STMTJRNL\_SPILL

\*\* <dd>^The SQLITE\_CONFIG\_STMTJRNL\_SPILL option takes a single parameter which

\*\* becomes the [statement journal] spill-to-disk threshold.

\*\* [Statement journals] are held in memory until their size (in bytes)

\*\* exceeds this threshold, at which point they are written to disk.

```

** Or if the threshold is -1, statement journals are always held
** exclusively in memory.
** Since many statement journals never become large, setting the spill
** threshold to a value such as 64KiB can greatly reduce the amount of
** I/O required to support statement rollback.
** The default value for this setting is controlled by the
** [SQLITE_STMTJRNL_SPILL] compile-time option.
**
** [[SQLITE_CONFIG_SORTERREF_SIZE]]
** <dt>SQLITE_CONFIG_SORTERREF_SIZE
** <dd>The SQLITE_CONFIG_SORTERREF_SIZE option accepts a single parameter
** of type (int) - the new value of the sorter-reference size threshold.
** Usually, when SQLite uses an external sort to order records according
** to an ORDER BY clause, all fields required by the caller are present in the
** sorted records. However, if SQLite determines based on the declared type
** of a table column that its values are likely to be very large - larger
** than the configured sorter-reference size threshold - then a reference
** is stored in each sorted record and the required column values loaded
** from the database as records are returned in sorted order. The default
** value for this option is to never use this optimization. Specifying a
** negative value for this option restores the default behaviour.
** This option is only available if SQLite is compiled with the
** [SQLITE_ENABLE_SORTER_REFERENCES] compile-time option.
**
** [[SQLITE_CONFIG_MEMDB_MAXSIZE]]
** <dt>SQLITE_CONFIG_MEMDB_MAXSIZE
** <dd>The SQLITE_CONFIG_MEMDB_MAXSIZE option accepts a single parameter
** [sqlite3_int64] parameter which is the default maximum size for an in-memory
** database created using [sqlite3_deserialize()]. This default maximum
** size can be adjusted up or down for individual databases using the
** [SQLITE_FCNTL_SIZE_LIMIT] [sqlite3_file_control|file-control]. If this
** configuration setting is never used, then the default maximum is determined
** by the [SQLITE_MEMDB_DEFAULT_MAXSIZE] compile-time option. If that
** compile-time option is not set, then the default maximum is 1073741824.
** </dd>
**/

```

Found in path(s):

```

* /opt/cola/permits/1541729670_1675313227.6905026/0/sqlite-amalgamation-3400100-zip/sqlite-amalgamation-3400100/sqlite3.h

```

No license file was found, but licenses were detected in source scan.

```

/* This will be more informative in a later version. */

```

Found in path(s):

```

* /opt/cola/permits/1541729670_1675313227.6905026/0/sqlite-amalgamation-3400100-zip/sqlite-amalgamation-3400100/shell.c

```

No license file was found, but licenses were detected in source scan.

```
/*
** CAPI3REF: Configuration Options
** KEYWORDS: {configuration option}
**
** These constants are the available integer configuration options that
** can be passed as the first argument to the [sqlite3_config()] interface.
**
** New configuration options may be added in future releases of SQLite.
** Existing configuration options might be discontinued. Applications
** should check the return code from [sqlite3_config()] to make sure that
** the call worked. The [sqlite3_config()] interface will return a
** non-zero [error code] if a discontinued or unsupported configuration option
** is invoked.
**
** <dl>
** [[SQLITE_CONFIG_SINGLETHREAD]] <dt>SQLITE_CONFIG_SINGLETHREAD</dt>
** <dd>There are no arguments to this option. ^This option sets the
** [threading mode] to Single-thread. In other words, it disables
** all mutexing and puts SQLite into a mode where it can only be used
** by a single thread. ^If SQLite is compiled with
** the [SQLITE_THREADSAFE | SQLITE_THREADSAFE=0] compile-time option then
** it is not possible to change the [threading mode] from its default
** value of Single-thread and so [sqlite3_config()] will return
** [SQLITE_ERROR] if called with the SQLITE_CONFIG_SINGLETHREAD
** configuration option.</dd>
**
** [[SQLITE_CONFIG_MULTITHREAD]] <dt>SQLITE_CONFIG_MULTITHREAD</dt>
** <dd>There are no arguments to this option. ^This option sets the
** [threading mode] to Multi-thread. In other words, it disables
** mutexing on [database connection] and [prepared statement] objects.
** The application is responsible for serializing access to
** [database connections] and [prepared statements]. But other mutexes
** are enabled so that SQLite will be safe to use in a multi-threaded
** environment as long as no two threads attempt to use the same
** [database connection] at the same time. ^If SQLite is compiled with
** the [SQLITE_THREADSAFE | SQLITE_THREADSAFE=0] compile-time option then
** it is not possible to set the Multi-thread [threading mode] and
** [sqlite3_config()] will return [SQLITE_ERROR] if called with the
** SQLITE_CONFIG_MULTITHREAD configuration option.</dd>
**
** [[SQLITE_CONFIG_SERIALIZED]] <dt>SQLITE_CONFIG_SERIALIZED</dt>
** <dd>There are no arguments to this option. ^This option sets the
** [threading mode] to Serialized. In other words, this option enables
** all mutexes including the recursive
** mutexes on [database connection] and [prepared statement] objects.
** In this mode (which is the default when SQLite is compiled with
```

\*\* [SQLITE\_THREADSAFE=1]) the SQLite library will itself serialize access  
 \*\* to [database connections] and [prepared statements] so that the  
 \*\* application is free to use the same [database connection] or the  
 \*\* same [prepared statement] in different threads at the same time.  
 \*\* ^If SQLite is compiled with  
 \*\* the [SQLITE\_THREADSAFE | SQLITE\_THREADSAFE=0] compile-time option then  
 \*\* it is not possible to set the Serialized [threading mode] and  
 \*\* [sqlite3\_config()] will return [SQLITE\_ERROR] if called with the  
 \*\* SQLITE\_CONFIG\_SERIALIZED configuration option.</dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_MALLOC]] <dt>SQLITE\_CONFIG\_MALLOC</dt>  
 \*\* <dd> ^(The SQLITE\_CONFIG\_MALLOC option takes a single argument which is  
 \*\* a pointer to an instance of the [sqlite3\_mem\_methods] structure.  
 \*\* The argument specifies  
 \*\* alternative low-level memory allocation routines to be used in place of  
 \*\* the memory allocation routines built into SQLite.)^ ^SQLite makes  
 \*\* its own private copy of the content of the [sqlite3\_mem\_methods] structure  
 \*\* before the [sqlite3\_config()] call returns.</dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_GETMALLOC]] <dt>SQLITE\_CONFIG\_GETMALLOC</dt>  
 \*\* <dd> ^(The SQLITE\_CONFIG\_GETMALLOC option takes a single argument which  
 \*\* is a pointer to an instance of the [sqlite3\_mem\_methods] structure.  
 \*\* The [sqlite3\_mem\_methods]  
 \*\* structure is filled with the currently defined memory allocation routines.)^  
 \*\* This option can be used to overload the default memory allocation  
 \*\* routines with a wrapper that simulations memory allocation failure or  
 \*\* tracks memory usage, for example. </dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_SMALL\_MALLOC]] <dt>SQLITE\_CONFIG\_SMALL\_MALLOC</dt>  
 \*\* <dd> ^The SQLITE\_CONFIG\_SMALL\_MALLOC option takes single argument of  
 \*\* type int, interpreted as a boolean, which if true provides a hint to  
 \*\* SQLite that it should avoid large memory allocations if possible.  
 \*\* SQLite will run faster if it is free to make large memory allocations,  
 \*\* but some application might prefer to run slower in exchange for  
 \*\* guarantees about memory fragmentation that are possible if large  
 \*\* allocations are avoided. This hint is normally off.  
 \*\* </dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_MEMSTATUS]] <dt>SQLITE\_CONFIG\_MEMSTATUS</dt>  
 \*\* <dd> ^The SQLITE\_CONFIG\_MEMSTATUS option takes single argument of type int,  
 \*\* interpreted as a boolean, which enables or disables the collection of  
 \*\* memory allocation statistics. ^(When memory allocation statistics are  
 \*\* disabled, the following SQLite interfaces become non-operational:  
 \*\* <ul>  
 \*\* <li> [sqlite3\_hard\_heap\_limit64()]  
 \*\* <li> [sqlite3\_memory\_used()]  
 \*\* <li> [sqlite3\_memory\_highwater()]  
 \*\* <li> [sqlite3\_soft\_heap\_limit64()]

```

** <li> [sqlite3_status64()]
** </ul>)^
** ^Memory allocation statistics are enabled by default unless SQLite is
** compiled with [SQLITE_DEFAULT_MEMSTATUS]=0 in which case memory
** allocation statistics are disabled by default.
** </dd>
**
** [[SQLITE_CONFIG_SCRATCH]] <dt>SQLITE_CONFIG_SCRATCH</dt>
** <dd> The SQLITE_CONFIG_SCRATCH option is no longer used.
** </dd>
**
** [[SQLITE_CONFIG_PAGECACHE]] <dt>SQLITE_CONFIG_PAGECACHE</dt>
** <dd> ^The SQLITE_CONFIG_PAGECACHE option specifies a memory pool
** that SQLite can use for the database page cache with the default page
** cache implementation.
** This configuration option is a no-op if an application-defined page
** cache implementation is loaded using the [SQLITE_CONFIG_PCACHE2].
** ^There are three arguments to SQLITE_CONFIG_PAGECACHE: A pointer to
** 8-byte aligned memory (pMem), the size of each page cache line (sz),
** and the number of cache lines (N).
** The sz argument should be the size of the largest database page
** (a power of two between 512 and 65536) plus some extra bytes for each
** page header. ^The number of extra bytes needed by the page header
** can be determined using [SQLITE_CONFIG_PCACHE_HDRSZ].
** ^It is harmless, apart from the wasted memory,
** for the sz parameter to be larger than necessary. The pMem
** argument must be either a NULL pointer or a pointer to an 8-byte
** aligned block of memory of at least sz*N bytes, otherwise
** subsequent behavior is undefined.
** ^When pMem is not NULL, SQLite will strive to use the memory provided
** to satisfy page cache needs, falling back to [sqlite3_malloc()] if
** a page cache line is larger than sz bytes or if all of the pMem buffer
** is exhausted.
** ^If pMem is NULL and N is non-zero, then each database connection
** does an initial bulk allocation for page cache memory
** from [sqlite3_malloc()] sufficient for N cache lines if N is positive or
** of -1024*N bytes if N is negative. . ^If additional
** page cache memory is needed beyond what is provided by the initial
** allocation, then SQLite goes to [sqlite3_malloc()] separately for each
** additional cache line. </dd>
**
** [[SQLITE_CONFIG_HEAP]] <dt>SQLITE_CONFIG_HEAP</dt>
** <dd> ^The SQLITE_CONFIG_HEAP option specifies a static memory buffer
** that SQLite will use for all of its dynamic memory allocation needs
** beyond those provided for by [SQLITE_CONFIG_PAGECACHE].
** ^The SQLITE_CONFIG_HEAP option is only available if SQLite is compiled
** with either [SQLITE_ENABLE_MEMSYS3] or [SQLITE_ENABLE_MEMSYS5] and returns
** [SQLITE_ERROR] if invoked otherwise.

```

\*\* ^There are three arguments to `SQLITE_CONFIG_HEAP`:  
 \*\* An 8-byte aligned pointer to the memory,  
 \*\* the number of bytes in the memory buffer, and the minimum allocation size.  
 \*\* ^If the first pointer (the memory pointer) is `NULL`, then SQLite reverts  
 \*\* to using its default memory allocator (the system `malloc()` implementation),  
 \*\* undoing any prior invocation of `[SQLITE_CONFIG_MALLOC]`. ^If the  
 \*\* memory pointer is not `NULL` then the alternative memory  
 \*\* allocator is engaged to handle all of SQLites memory allocation needs.  
 \*\* The first pointer (the memory pointer) must be aligned to an 8-byte  
 \*\* boundary or subsequent behavior of SQLite will be undefined.  
 \*\* The minimum allocation size is capped at  $2^{12}$ . Reasonable values  
 \*\* for the minimum allocation size are  $2^5$  through  $2^8$ .</dd>  
 \*\*  
 \*\* `[SQLITE_CONFIG_MUTEX]` <dt>`SQLITE_CONFIG_MUTEX`</dt>  
 \*\* <dd> ^The `SQLITE_CONFIG_MUTEX` option takes a single argument which is a  
 \*\* pointer to an instance of the `[sqlite3_mutex_methods]` structure.  
 \*\* The argument specifies alternative low-level mutex routines to be used  
 \*\* in place the mutex routines built into SQLite.)^ ^SQLite makes a copy of  
 \*\* the content of the `[sqlite3_mutex_methods]` structure before the call to  
 \*\* `[sqlite3_config()]` returns. ^If SQLite is compiled with  
 \*\* the `[SQLITE_THREADSAFE | SQLITE_THREADSAFE=0]` compile-time option then  
 \*\* the entire mutexing subsystem is omitted from the build and hence calls to  
 \*\* `[sqlite3_config()]` with the `SQLITE_CONFIG_MUTEX` configuration option will  
 \*\* return `[SQLITE_ERROR]`.</dd>  
 \*\*  
 \*\* `[SQLITE_CONFIG_GETMUTEX]` <dt>`SQLITE_CONFIG_GETMUTEX`</dt>  
 \*\* <dd> ^The `SQLITE_CONFIG_GETMUTEX` option takes a single argument which  
 \*\* is a pointer to an instance of the `[sqlite3_mutex_methods]` structure. The  
 \*\* `[sqlite3_mutex_methods]`  
 \*\* structure is filled with the currently defined mutex routines.)^  
 \*\* This option can be used to overload the default mutex allocation  
 \*\* routines with a wrapper used to track mutex usage for performance  
 \*\* profiling or testing, for example. ^If SQLite is compiled with  
 \*\* the `[SQLITE_THREADSAFE | SQLITE_THREADSAFE=0]` compile-time option then  
 \*\* the entire mutexing subsystem is omitted from the build and hence calls to  
 \*\* `[sqlite3_config()]` with the `SQLITE_CONFIG_GETMUTEX` configuration option will  
 \*\* return `[SQLITE_ERROR]`.</dd>  
 \*\*  
 \*\* `[SQLITE_CONFIG_LOOKASIDE]` <dt>`SQLITE_CONFIG_LOOKASIDE`</dt>  
 \*\* <dd> ^The `SQLITE_CONFIG_LOOKASIDE` option takes two arguments that determine  
 \*\* the default size of lookaside memory on each [database connection].  
 \*\* The first argument is the  
 \*\* size of each lookaside buffer slot and the second is the number of  
 \*\* slots allocated to each database connection.)^ ^(`SQLITE_CONFIG_LOOKASIDE`  
 \*\* sets the <i>default</i> lookaside size. The `[SQLITE_DBCONFIG_LOOKASIDE]`  
 \*\* option to `[sqlite3_db_config()]` can be used to change the lookaside  
 \*\* configuration on individual connections.)^ </dd>  
 \*\*

**SQLITE\_CONFIG\_PCACHE2** `<dt>SQLITE_CONFIG_PCACHE2</dt>`  
`<dd>` `^(The SQLITE_CONFIG_PCACHE2 option takes a single argument which is`  
`a pointer to an [sqlite3_pcache_methods2] object. This object specifies`  
`the interface to a custom page cache implementation.)^`  
`^SQLite makes a copy of the [sqlite3_pcache_methods2] object.</dd>`

**SQLITE\_CONFIG\_GETPCACHE2** `<dt>SQLITE_CONFIG_GETPCACHE2</dt>`  
`<dd>` `^(The SQLITE_CONFIG_GETPCACHE2 option takes a single argument which`  
`is a pointer to an [sqlite3_pcache_methods2] object. SQLite copies of`  
`the current page cache implementation into that object.)^ </dd>`

**SQLITE\_CONFIG\_LOG** `<dt>SQLITE_CONFIG_LOG</dt>`  
`<dd>` The `SQLITE_CONFIG_LOG` option is used to configure the SQLite  
global [error log].  
`^(The SQLITE_CONFIG_LOG option takes two arguments: a pointer to a`  
`function with a call signature of void (*)(void*,int,const char*),`  
`and a pointer to void. ^If the function pointer is not NULL, it is`  
`invoked by [sqlite3_log()] to process each logging event. ^If the`  
`function pointer is NULL, the [sqlite3_log()] interface becomes a no-op.`  
`^The void pointer that is the second argument to SQLITE_CONFIG_LOG is`  
`passed through as the first parameter to the application-defined logger`  
`function whenever that function is invoked. ^The second parameter to`  
`the logger function is a copy of the first parameter to the corresponding`  
`[sqlite3_log()] call and is intended to be a [result code] or an`  
`[extended result code]. ^The third parameter passed to the logger is`  
`log message after formatting via [sqlite3_snprintf()].`  
`The SQLite logging interface is not reentrant; the logger function`  
`supplied by the application must not invoke any SQLite interface.`  
`In a multi-threaded application, the application-defined logger`  
`function must be threadsafe. </dd>`

**SQLITE\_CONFIG\_URI** `<dt>SQLITE_CONFIG_URI`  
`<dd>``^(The SQLITE_CONFIG_URI option takes a single argument of type int.`  
`If non-zero, then URI handling is globally enabled. If the parameter is zero,`  
`then URI handling is globally disabled.)^ ^If URI handling is globally`  
`enabled, all filenames passed to [sqlite3_open()], [sqlite3_open_v2()],`  
`[sqlite3_open16()] or`  
`specified as part of [ATTACH] commands are interpreted as URIs, regardless`  
`of whether or not the [SQLITE_OPEN_URI] flag is set when the database`  
`connection is opened. ^If it is globally disabled, filenames are`  
`only interpreted as URIs if the SQLITE_OPEN_URI flag is set when the`  
`database connection is opened. ^(By default, URI handling is globally`  
`disabled. The default value may be changed by compiling with the`  
`[SQLITE_USE_URI] symbol defined.)^`

**SQLITE\_CONFIG\_COVERING\_INDEX\_SCAN** `<dt>SQLITE_CONFIG_COVERING_INDEX_SCAN`  
`<dd>``^The SQLITE_CONFIG_COVERING_INDEX_SCAN option takes a single integer`  
`argument which is interpreted as a boolean in order to enable or disable`

\*\* the use of covering indices for full table scans in the query optimizer.  
 \*\* ^The default setting is determined  
 \*\* by the [SQLITE\_ALLOW\_COVERING\_INDEX\_SCAN] compile-time option, or is "on"  
 \*\* if that compile-time option is omitted.  
 \*\* The ability to disable the use of covering indices for full table scans  
 \*\* is because some incorrectly coded legacy applications might malfunction  
 \*\* when the optimization is enabled. Providing the ability to  
 \*\* disable the optimization allows the older, buggy application code to work  
 \*\* without change even with newer versions of SQLite.  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_PCACHE]] [[SQLITE\_CONFIG\_GETPCACHE]]  
 \*\* <dt>SQLITE\_CONFIG\_PCACHE and SQLITE\_CONFIG\_GETPCACHE  
 \*\* <dd> These options are obsolete and should not be used by new code.  
 \*\* They are retained for backwards compatibility but are now no-ops.  
 \*\* </dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_SQLLOG]]  
 \*\* <dt>SQLITE\_CONFIG\_SQLLOG  
 \*\* <dd>This option is only available if sqlite is compiled with the  
 \*\* [SQLITE\_ENABLE\_SQLLOG] pre-processor macro defined. The first argument should  
 \*\* be a pointer to a function of type void(\*)(void\*,sqlite3\*,const char\*, int).  
 \*\* The second should be of type (void\*). The callback is invoked by the library  
 \*\* in three separate circumstances, identified by the value passed as the  
 \*\* fourth parameter. If the fourth parameter is 0, then the database connection  
 \*\* passed as the second argument has just been opened. The third argument  
 \*\* points to a buffer containing the name of the main database file. If the  
 \*\* fourth parameter is 1, then the SQL statement that the third parameter  
 \*\* points to has just been executed. Or, if the fourth parameter is 2, then  
 \*\* the connection being passed as the second parameter is being closed. The  
 \*\* third parameter is passed NULL In this case. An example of using this  
 \*\* configuration option can be seen in the "test\_sqllog.c" source file in  
 \*\* the canonical SQLite source tree.</dd>  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_MMAP\_SIZE]]  
 \*\* <dt>SQLITE\_CONFIG\_MMAP\_SIZE  
 \*\* <dd>^SQLITE\_CONFIG\_MMAP\_SIZE takes two 64-bit integer (sqlite3\_int64) values  
 \*\* that are the default mmap size limit (the default setting for  
 \*\* [PRAGMA mmap\_size]) and the maximum allowed mmap size limit.  
 \*\* ^The default setting can be overridden by each database connection using  
 \*\* either the [PRAGMA mmap\_size] command, or by using the  
 \*\* [SQLITE\_FCNTL\_MMAP\_SIZE] file control. ^(The maximum allowed mmap size  
 \*\* will be silently truncated if necessary so that it does not exceed the  
 \*\* compile-time maximum mmap size set by the  
 \*\* [SQLITE\_MAX\_MMAP\_SIZE] compile-time option.)^  
 \*\* ^If either argument to this option is negative, then that argument is  
 \*\* changed to its compile-time default.  
 \*\*  
 \*\* [[SQLITE\_CONFIG\_WIN32\_HEAPSIZE]]

```

** <dt>SQLITE_CONFIG_WIN32_HEAPSIZE
** <dd>^The SQLITE_CONFIG_WIN32_HEAPSIZE option is only available if SQLite is
** compiled for Windows with the [SQLITE_WIN32_MALLOC] pre-processor macro
** defined. ^SQLITE_CONFIG_WIN32_HEAPSIZE takes a 32-bit unsigned integer value
** that specifies the maximum size of the created heap.
**
** [[SQLITE_CONFIG_PCACHE_HDRSZ]]
** <dt>SQLITE_CONFIG_PCACHE_HDRSZ
** <dd>^The SQLITE_CONFIG_PCACHE_HDRSZ option takes a single parameter which
** is a pointer to an integer and writes into that integer the number of extra
** bytes per page required for each page in [SQLITE_CONFIG_PAGECACHE].
** The amount of extra space required can change depending on the compiler,
** target platform, and SQLite version.
**
** [[SQLITE_CONFIG_PMASZ]]
** <dt>SQLITE_CONFIG_PMASZ
** <dd>^The SQLITE_CONFIG_PMASZ option takes a single parameter which
** is an unsigned integer and sets the "Minimum PMA Size" for the multithreaded
** sorter to that integer. The default minimum PMA Size is set by the
** [SQLITE_SORTER_PMASZ] compile-time option. New threads are launched
** to help with sort operations when multithreaded sorting
** is enabled (using the [PRAGMA threads] command) and the amount of content
** to be sorted exceeds the page size times the minimum of the
** [PRAGMA cache_size] setting and this value.
**
** [[SQLITE_CONFIG_STMTJRNL_SPILL]]
** <dt>SQLITE_CONFIG_STMTJRNL_SPILL
** <dd>^The SQLITE_CONFIG_STMTJRNL_SPILL option takes a single parameter which
** becomes the [statement journal] spill-to-disk threshold.
** [Statement journals] are held in memory until their size (in bytes)
** exceeds this threshold, at which point they are written to disk.
** Or if the threshold is -1, statement journals are always held
** exclusively in memory.
** Since many statement journals never become large, setting the spill
** threshold to a value such as 64KiB can greatly reduce the amount of
** I/O required to support statement rollback.
** The default value for this setting is controlled by the
** [SQLITE_STMTJRNL_SPILL] compile-time option.
**
** [[SQLITE_CONFIG_SORTERREF_SIZE]]
** <dt>SQLITE_CONFIG_SORTERREF_SIZE
** <dd>The SQLITE_CONFIG_SORTERREF_SIZE option accepts a single parameter
** of type (int) - the new value of the sorter-reference size threshold.
** Usually, when SQLite uses an external sort to order records according
** to an ORDER BY clause, all fields required by the caller are present in the
** sorted records. However, if SQLite determines based on the declared type
** of a table column that its values are likely to be very large - larger
** than the configured sorter-reference size threshold - then a reference

```

```

** is stored in each sorted record and the required column values loaded
** from the database as records are returned in sorted order. The default
** value for this option is to never use this optimization. Specifying a
** negative value for this option restores the default behaviour.
** This option is only available if SQLite is compiled with the
** [SQLITE_ENABLE_SORTER_REFERENCES] compile-time option.
**
** [[SQLITE_CONFIG_MEMDB_MAXSIZE]]
** <dt>SQLITE_CONFIG_MEMDB_MAXSIZE
** <dd>The SQLITE_CONFIG_MEMDB_MAXSIZE option accepts a single parameter
** [sqlite3_int64] parameter which is the default maximum size for an in-memory
** database created using [sqlite3_deserialize()]. This default maximum
** size can be adjusted up or down for individual databases using the
** [SQLITE_FCNTL_SIZE_LIMIT] [sqlite3_file_control|file-control]. If this
** configuration setting is never used, then the default maximum is determined
** by the [SQLITE_MEMDB_DEFAULT_MAXSIZE] compile-time option. If that
** compile-time option is not set, then the default maximum is 1073741824.
** </dd>
** /
** /*
** The "printf" code that follows dates from the 1980's. It is in
** the public domain.
**
** *****
**
** This file contains code for a set of "printf"-like routines. These
** routines format strings much like the printf() from the standard C
** library, though the implementation here has enhancements to support
** SQLite.
** /
** /*
** 2004 May 22
**
** The author disclaims copyright to this source code. In place of
** a legal notice, here is a blessing:
**
** May you do good and not evil.
** May you find forgiveness for yourself and forgive others.
** May you share freely, never taking more than you give.
**
** *****
**
** This file contains the VFS implementation for unix-like operating systems
** include Linux, MacOSX, *BSD, QNX, VxWorks, AIX, HPUX, and others.
**
** There are actually several different VFS implementations in this file.
** The differences are in the way that file locking is done. The default
** implementation uses Posix Advisory Locks. Alternative implementations

```

```

** use flock(), dot-files, various proprietary locking schemas, or simply
** skip locking all together.
**
** This source file is organized into divisions where the logic for various
** subfunctions is contained within the appropriate division. PLEASE
** KEEP THE STRUCTURE OF THIS FILE INTACT. New code should be placed
** in the correct division and should be clearly labeled.
**
** The layout of divisions is as follows:
**
** * General-purpose declarations and utility functions.
** * Unique file ID logic used by VxWorks.
** * Various locking primitive implementations (all except proxy locking):
**   + for Posix Advisory Locks
**   + for no-op locks
**   + for dot-file locks
**   + for flock() locking
**   + for named semaphore locks (VxWorks only)
**   + for AFP filesystem locks (MacOSX only)
** * sqlite3_file methods not associated with locking.
** * Definitions of sqlite3_io_methods objects for all locking
**   methods plus "finder" functions for each locking method.
** * sqlite3_vfs method implementations.
** * Locking primitives for the proxy uber-locking-method. (MacOSX only)
** * Definitions of sqlite3_vfs objects for all locking methods
**   plus implementations of sqlite3_os_init() and sqlite3_os_end().
**
**/
/*
** Return a pointer to the "temporary page" buffer held internally
** by the pager. This is a buffer that is big enough to hold the
** entire content of a database page. This buffer is used internally
** during rollback and will be overwritten whenever a rollback
** occurs. But other modules are free to use it too, as long as
** no rollbacks are happening.
**/

```

Found in path(s):

```

* /opt/cola/permits/1541729670_1675313227.6905026/0/sqlite-amalgamation-3400100-zip/sqlite-amalgamation-3400100/sqlite3.c

```

## 1.11 openssl 1.1.1k

### 1.11.1 Notifications :

This product includes software written by Tim Hudson (tjh@cryptsoft.com).

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit  
(<http://www.openssl.org/>)

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com).

## 1.11.2 Available under license :

### LICENSE ISSUES

=====

The OpenSSL toolkit stays under a double license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts.

#### OpenSSL License

-----

```
/* =====  
* Copyright (c) 1998-2019 The OpenSSL Project. All rights reserved.  
*  
* Redistribution and use in source and binary forms, with or without  
* modification, are permitted provided that the following conditions  
* are met:  
*  
* 1. Redistributions of source code must retain the above copyright  
* notice, this list of conditions and the following disclaimer.  
*  
* 2. Redistributions in binary form must reproduce the above copyright  
* notice, this list of conditions and the following disclaimer in  
* the documentation and/or other materials provided with the  
* distribution.  
*  
* 3. All advertising materials mentioning features or use of this  
* software must display the following acknowledgment:  
* "This product includes software developed by the OpenSSL Project  
* for use in the OpenSSL Toolkit. (http://www.openssl.org/)"  
*  
* 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to  
* endorse or promote products derived from this software without  
* prior written permission. For written permission, please contact  
* openssl-core@openssl.org.  
*  
* 5. Products derived from this software may not be called "OpenSSL"  
* nor may "OpenSSL" appear in their names without prior written  
* permission of the OpenSSL Project.  
*  
* 6. Redistributions of any form whatsoever must retain the following  
* acknowledgment:  
* "This product includes software developed by the OpenSSL Project  
* for use in the OpenSSL Toolkit (http://www.openssl.org/)"  
*  
* THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY
```

\* EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE  
\* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR  
\* PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR  
\* ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,  
\* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT  
\* NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;  
\* LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)  
\* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,  
\* STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)  
\* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED  
\* OF THE POSSIBILITY OF SUCH DAMAGE.

\* =====

\*

\* This product includes cryptographic software written by Eric Young  
\* (eay@cryptsoft.com). This product includes software written by Tim  
\* Hudson (tjh@cryptsoft.com).

\*

\*/

#### Original SSLeay License

-----

/\* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)

\* All rights reserved.

\*

\* This package is an SSL implementation written

\* by Eric Young (eay@cryptsoft.com).

\* The implementation was written so as to conform with Netscapes SSL.

\*

\* This library is free for commercial and non-commercial use as long as

\* the following conditions are aheared to. The following conditions

\* apply to all code found in this distribution, be it the RC4, RSA,

\* lhash, DES, etc., code; not just the SSL code. The SSL documentation

\* included with this distribution is covered by the same copyright terms

\* except that the holder is Tim Hudson (tjh@cryptsoft.com).

\*

\* Copyright remains Eric Young's, and as such any Copyright notices in

\* the code are not to be removed.

\* If this package is used in a product, Eric Young should be given attribution

\* as the author of the parts of the library used.

\* This can be in the form of a textual message at program startup or

\* in documentation (online or textual) provided with the package.

\*

\* Redistribution and use in source and binary forms, with or without

\* modification, are permitted provided that the following conditions

\* are met:

\* 1. Redistributions of source code must retain the copyright

\* notice, this list of conditions and the following disclaimer.

- \* 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- \* 3. All advertising materials mentioning features or use of this software must display the following acknowledgement:
  - \* "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)"
  - \* The word 'cryptographic' can be left out if the routines from the library being used are not cryptographic related :-).
- \* 4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement:
  - \* "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"
- \* THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
- \* The licence and distribution terms for any publically available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution licence [including the GNU Public Licence.]
- \*/

## 1.12 libcxxabi 9.0.9svn

### 1.12.1 Available under license :

```
=====
libc++abi License
=====
```

The libc++abi library is dual licensed under both the University of Illinois "BSD-Like" license and the MIT license. As a user of this code you may choose to use it under either license. As a contributor, you agree to allow your code to be used under both.

Full text of the relevant licenses is included below.

```
=====
```

University of Illinois/NCSA  
Open Source License

Copyright (c) 2009-2018 by the contributors listed in CREDITS.TXT

All rights reserved.

Developed by:

LLVM Team

University of Illinois at Urbana-Champaign

<http://llvm.org>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal with the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

- \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimers.
- \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimers in the documentation and/or other materials provided with the distribution.
- \* Neither the names of the LLVM Team, University of Illinois at Urbana-Champaign, nor the names of its contributors may be used to endorse or promote products derived from this Software without specific prior written permission.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE CONTRIBUTORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS WITH THE SOFTWARE.

=====  
Copyright (c) 2009-2014 by the contributors listed in CREDITS.TXT

Permission is hereby granted, free of charge, to any person obtaining a copy

of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

=====  
libc++abi License  
=====

The libc++abi library is dual licensed under both the University of Illinois "BSD-Like" license and the MIT license. As a user of this code you may choose to use it under either license. As a contributor, you agree to allow your code to be used under both.

Full text of the relevant licenses is included below.

=====  
University of Illinois/NCSA  
Open Source License

Copyright (c) 2009-2014 by the contributors listed in CREDITS.TXT

All rights reserved.

Developed by:

LLVM Team

University of Illinois at Urbana-Champaign

<http://llvm.org>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal with the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies

of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

- \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimers.
- \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimers in the documentation and/or other materials provided with the distribution.
- \* Neither the names of the LLVM Team, University of Illinois at Urbana-Champaign, nor the names of its contributors may be used to endorse or promote products derived from this Software without specific prior written permission.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE CONTRIBUTORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS WITH THE SOFTWARE.

=====

Copyright (c) 2009-2014 by the contributors listed in CREDITS.TXT

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 1.13 crashlytics 2.5.2

## 1.13.1 Available under license :

SDK:

protobuf-c <http://code.google.com/p/protobuf-c/source/browse/trunk/LICENSE>

Copyright (c) 2008-2011, Dave Benson.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

\* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

\* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

\* Neither the name of "protobuf-c" nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Square Tape - QueueFile.java <https://raw.githubusercontent.com/square/tape/master/LICENSE.txt>

Copyright 2012 Square, Inc.

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

This file has been modified from its original licensed form.

Modifications are Copyright (C) 2013 Twitter, Inc.

Apache Software Foundation <http://www.apache.org/licenses/LICENSE-2.0>

Apache Commons CLI

Copyright 2001-2013 The Apache Software Foundation

Apache Commons Codec

Copyright 2002-2013 The Apache Software Foundation

Apache Felix Copyright

2006-2013 The Apache Software Foundation

Apache HttpComponents Client

Copyright 1999-2013 The Apache Software Foundation

Apache HttpComponents Core

Copyright 1999-2013 The Apache Software Foundation

Apache log4j

Copyright 2010-2013 The Apache Software Foundation

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

JSON.simple <http://code.google.com/p/json-simple/source/browse/trunk/LICENSE.txt>

Copyright 2006-2012 Yidong Fang Chris Nokleberg Dave Hughes

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.

SLF4J <http://www.slf4j.org/license.html>

Copyright (c) 2004-2013 QOS.ch  
All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining  
a copy of this software and associated documentation files (the  
"Software"), to deal in the Software without restriction, including  
without limitation the rights to use, copy, modify, merge, publish,  
distribute, sublicense, and/or sell copies of the Software, and to  
permit persons to whom the Software is furnished to do so, subject to  
the following conditions:

The above copyright notice and this permission notice shall be  
included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,  
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF  
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE  
LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION  
OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION  
WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Timing Framwork <https://java.net/projects/timingframework>

Copyright (c) 2013, SureLogic, Inc

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Otto - <https://raw.githubusercontent.com/square/otto/master/LICENSE.txt>

Copyright 2013 Square, Inc.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Beta By Crashlytics App:

Android Open Source Project <https://source.android.com/>

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

ASM Core <http://asm.ow2.org/>

Copyright (c) 2000-2011 INRIA, France Telecom  
All rights reserved.

Redistribution and use in source and binary forms, with or without

modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holders nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Auto <https://github.com/google/auto>

Copyright 2013 Google, Inc.

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

ButterKnife <http://jakewharton.github.io/butterknife/>

Copyright 2013 Jake Wharton

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.

Dagger: <http://square.github.io/dagger/>

Copyright 2013 Square, Inc.

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.

Guava <https://github.com/google/guava>

Copyright (C) 2010 The Guava Authors

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.

JavaWriter [https://github.com/square/javapoet/tree/javawriter\\_2](https://github.com/square/javapoet/tree/javawriter_2)

Copyright 2013 Square, Inc.

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.

Nine Old Androids <https://github.com/JakeWharton/NineOldAndroids>

Copyright 2012 Jake Wharton

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.

Material Dialogs <https://github.com/afollestad/material-dialogs>

Copyright (c) 2015 Aidan Michael Follestad

Permission is hereby granted, free of charge, to any person obtaining a copy  
of this software and associated documentation files (the "Software"), to deal  
in the Software without restriction, including without limitation the rights  
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
copies of the Software, and to permit persons to whom the Software is  
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all  
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR

IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Otto <http://square.github.io/otto/>

Copyright 2013 Square, Inc.

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Picasso <http://square.github.io/picasso/>

Copyright 2013 Square, Inc.

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

RxAndroid <https://github.com/ReactiveX/RxAndroid>

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

RxJava <https://github.com/ReactiveX/RxJava>

Copyright 2013 Netflix, Inc.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Timber: <https://github.com/JakeWharton/timber>

Copyright 2013 Jake Wharton

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

JSR-250 Common Annotations for the Java™ Platform:

<http://jcp.org/aboutJava/communityprocess/final/jsr250/index.html>

COMMON DEVELOPMENT AND DISTRIBUTION LICENSE (CDDL) Version 1.0 1.

Definitions.

- 1.1. Contributor means each individual or entity that creates or contributes to the creation of Modifications.
- 1.2. Contributor Version means the combination of the Original Software, prior Modifications used by a Contributor (if any), and the Modifications made by that particular Contributor.
- 1.3. Covered Software means (a) the Original Software, or (b) Modifications, or (c) the combination of files containing Original Software with files containing Modifications, in each case including portions thereof.
- 1.4. Executable means the Covered Software in any form other than Source Code.
- 1.5. Initial Developer means the individual or entity that first makes Original Software available under this License.
- 1.6. Larger Work means a work which combines Covered Software or portions thereof with code not governed by the terms of this License.
- 1.7. License means this document.
- 1.8. Licensable means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently acquired, any and all of the rights conveyed herein.
- 1.9. Modifications means the Source Code and Executable form of any of the following: A. Any file that results from an addition to, deletion from or modification of the contents of a file containing Original Software or previous Modifications; B. Any new file that contains any part of the Original Software or previous Modification; or C. Any new file that is contributed or otherwise made available under the terms of this License.
- 1.10. Original Software means the Source Code and Executable form of computer software code that is originally released under this License.
- 1.11. Patent Claims means any patent claim(s), now owned or hereafter acquired, including without limitation, method, process, and apparatus claims, in any patent Licensable by grantor.
- 1.12. Source Code means (a) the common form of computer software code in which modifications are made and (b) associated documentation included in or with such code.
- 1.13. You (or Your) means an individual or a legal entity exercising rights under, and complying with all of the terms of, this License. For legal entities, You includes any entity which controls, is controlled by, or is under common control with You. For purposes of this definition, control means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity.

## 2. License Grants.

2.1. The Initial Developer Grant. Conditioned upon Your compliance with Section 3.1 below and subject to third party intellectual property claims, the Initial Developer hereby grants You a world-wide, royalty-free, non-exclusive license:

- (a) under intellectual property rights (other than patent or trademark) Licensable by Initial Developer, to use,

reproduce, modify, display, perform, sublicense and distribute the Original Software (or portions thereof), with or without Modifications, and/or as part of a Larger Work; and

(b) under Patent Claims infringed by the making, using or selling of Original Software, to make, have made, use, practice, sell, and offer for sale, and/or otherwise dispose of the Original Software (or portions thereof);

(c) The licenses granted in Sections 2.1(a) and (b) are effective on the date Initial Developer first distributes or otherwise makes the Original Software available to a third party under the terms of this License;

(d) Notwithstanding Section 2.1(b) above, no patent license is granted: (1) for code that You delete from the Original Software, or (2) for infringements caused by: (i) the modification of the Original Software, or (ii) the combination of the Original Software with other software or devices.

2.2. Contributor Grant. Conditioned upon Your compliance with Section 3.1 below and subject to third party intellectual property claims, each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license:

(a) under intellectual property rights (other than patent or trademark) Licensable by Contributor to use, reproduce, modify, display, perform, sublicense and distribute the Modifications created by such Contributor (or portions thereof), either on an unmodified basis, with other Modifications, as Covered Software and/or as part of a Larger Work; and

(b) under Patent Claims infringed by the making, using, or selling of Modifications made by that Contributor either alone and/or in combination with its Contributor Version (or portions of such combination), to make, use, sell, offer for sale, have made, and/or otherwise dispose of: (1) Modifications made by that Contributor (or portions thereof); and (2) the combination of Modifications made by that Contributor with its Contributor Version (or portions of such combination).

(c) The licenses granted in Sections 2.2(a) and 2.2(b) are effective on the date Contributor first distributes or otherwise makes the Modifications available to a third party.

(d) Notwithstanding Section 2.2(b) above, no patent license is granted: (1) for any code that Contributor has deleted from the Contributor Version; (2) for infringements caused by: (i) third party modifications of Contributor Version, or (ii) the combination of Modifications made by that Contributor with other software (except as part of the Contributor Version) or other devices; or (3) under Patent Claims infringed by Covered Software in the absence of Modifications made by that Contributor.

### 3. Distribution Obligations.

3.1. Availability of Source Code. Any Covered Software that You distribute or otherwise make available in Executable form must also be made available in Source Code form and that Source Code form must be distributed only under the terms of this License. You must include a copy of this License with every copy of the Source Code form of the Covered Software You distribute or otherwise make available. You must inform recipients of any such Covered Software in Executable form as to how they can obtain such Covered Software in Source Code form in a reasonable manner on or through a medium customarily used for software exchange.

3.2. Modifications. The Modifications that You create or to which You contribute are governed by the terms of this License. You represent that You believe Your Modifications are Your original creation(s) and/or You have sufficient rights to grant the rights conveyed by this License.

3.3. Required Notices. You must include a notice in each of Your Modifications that identifies You as the Contributor of the Modification. You may not remove or alter any copyright, patent or trademark notices contained within the Covered Software, or any notices of licensing or any descriptive text giving attribution to any Contributor or the Initial Developer.

3.4. Application of Additional Terms. You may not offer or impose any terms on any Covered Software in Source Code form that alters or restricts the applicable version of this License or the recipients rights hereunder. You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Software. However, you may do so only on Your own behalf, and not on behalf of the Initial Developer or any Contributor. You must make it absolutely clear that any such warranty, support, indemnity or liability obligation is offered by You alone, and You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of warranty, support, indemnity or liability terms You offer.

3.5. Distribution of Executable Versions. You may distribute the Executable form of the Covered Software under the terms of this License or under the terms of a license of Your choice, which may contain terms different from this License, provided that You are in compliance with the terms of this License and that the license for the Executable form does not attempt to limit or alter the recipients rights in the Source Code form from the rights set forth in this License. If You distribute the Covered Software in Executable form under a different license, You must make it absolutely clear that any terms which differ from this License are offered by You alone, not by the Initial Developer or Contributor. You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of any such terms You offer.

3.6. Larger Works. You may create a Larger Work by combining Covered Software with other code not governed by the terms of this License and distribute the Larger Work as a single product. In such a case, You must make sure the requirements of this License are fulfilled for the Covered Software.

#### 4. Versions of the License.

4.1. New Versions. Sun Microsystems, Inc. is the initial license steward and may publish revised and/or new versions of this License from time to time. Each version will be given a distinguishing version number. Except as provided in Section 4.3, no one other than the license steward has the right to modify this License.

4.2. Effect of New Versions. You may always continue to use, distribute or otherwise make the Covered Software available under the terms of the version of the License under which You originally received the Covered Software. If the Initial Developer includes a notice in the Original Software prohibiting it from being distributed or otherwise made available under any subsequent version of the License, You must distribute and make the Covered Software available under the terms of the version of the License under which You originally received the Covered Software. Otherwise, You may also choose to use, distribute or otherwise make the Covered Software available under the terms of any subsequent version of the License published by the license steward.

4.3. Modified Versions. When You are an Initial Developer and You want to create a new license for Your Original Software, You may create and use a modified version of this License if You: (a) rename the license and remove any references to the name of the license steward (except to note that the license differs from this License); and (b) otherwise make it clear that the license contains terms which differ from this License.

5. DISCLAIMER OF WARRANTY. COVERED SOFTWARE IS PROVIDED UNDER THIS LICENSE ON AN

AS IS BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, WARRANTIES THAT THE COVERED SOFTWARE IS FREE OF DEFECTS, MERCHANTABILITY, FIT FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE COVERED SOFTWARE IS WITH YOU. SHOULD ANY COVERED SOFTWARE PROVE DEFECTIVE IN ANY RESPECT, YOU (NOT THE INITIAL DEVELOPER OR ANY OTHER CONTRIBUTOR) ASSUME THE COST OF ANY NECESSARY SERVICING, REPAIR OR CORRECTION. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF ANY COVERED SOFTWARE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

## 6. TERMINATION.

6.1. This License and the rights granted hereunder will terminate automatically if You fail to comply with terms herein and fail to cure such breach within 30 days of becoming aware of the breach. Provisions which, by their nature, must remain in effect beyond the termination of this License shall survive.

6.2. If You assert a patent infringement claim (excluding declaratory judgment actions) against Initial Developer or a Contributor (the Initial Developer or Contributor against whom You assert such claim is referred to as Participant) alleging that the Participant Software (meaning the Contributor Version where the Participant is a Contributor or the Original Software where the Participant is the Initial Developer) directly or indirectly infringes any patent, then any and all rights granted directly or indirectly to You by such Participant, the Initial Developer (if the Initial Developer is not the Participant) and all Contributors under Sections 2.1 and/or 2.2 of this License shall, upon 60 days notice from Participant terminate prospectively and automatically at the expiration of such 60 day notice period, unless if within such 60 day period You withdraw Your claim with respect to the Participant Software against such Participant either unilaterally or pursuant to a written agreement with Participant.

6.3. In the event of termination under Sections 6.1 or 6.2 above, all end user licenses that have been validly granted by You or any distributor hereunder prior to termination (excluding licenses granted to You by any distributor) shall survive termination.

7. LIMITATION OF LIABILITY. UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL YOU, THE INITIAL DEVELOPER, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF COVERED SOFTWARE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO ANY PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOST PROFITS, LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY RESULTING FROM SUCH PARTYS NEGLIGENCE TO THE EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU.

8. U.S. GOVERNMENT END USERS. The Covered Software is a commercial item, as that term is defined in 48 C.F.R. 2.101 (Oct. 1995), consisting of commercial computer software (as that term is defined at 48 C.F.R. 252.227-7014(a)(1)) and commercial computer software documentation as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S.

Government End Users acquire Covered Software with only those rights set forth herein. This U.S. Government Rights clause is in lieu of, and supersedes, any other FAR, DFAR, or other clause or provision that addresses Government rights in computer software under this License.

9. MISCELLANEOUS. This License represents the complete agreement concerning subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. This License shall be governed by the law of the jurisdiction specified in a notice contained within the Original Software (except to the extent applicable law, if any, provides otherwise), excluding such jurisdictions conflict-of-law provisions. Any litigation relating to this License shall be subject to the jurisdiction of the courts located in the jurisdiction and venue specified in a notice contained within the Original Software, with the losing party responsible for costs, including, without limitation, court costs and reasonable attorneys fees and expenses. The application of the United Nations Convention on Contracts for the International Sale of Goods is expressly excluded. Any law or regulation which provides that the language of a contract shall be construed against the drafter shall not apply to this License. You agree that You alone are responsible for compliance with the United States export administration regulations (and the export control laws and regulation of any other countries) when You use, distribute or otherwise make available any Covered Software.

10. RESPONSIBILITY FOR CLAIMS. As between Initial Developer and the Contributors, each party is responsible for claims and damages arising, directly or indirectly, out of its utilization of rights under this License and You agree to work with Initial Developer and Contributors to distribute such responsibility on an equitable basis. Nothing herein is intended or shall be deemed to constitute any admission of liability.

NOTICE PURSUANT TO SECTION 9 OF THE COMMON DEVELOPMENT AND DISTRIBUTION LICENSE (CDDL) The code released under the CDDL shall be governed by the laws of the State of California (excluding conflict-of-law provisions). Any litigation relating to this License shall be subject to the jurisdiction of the Federal Courts of the Northern District of California and the state courts of the State of California, with venue lying in Santa Clara County, California.

## 1.14 libilbc 2.0.2

### 1.14.1 Available under license :

Copyright (c) 2011, The WebRTC project authors. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- \* Neither the name of Google nor the names of its contributors may be used to endorse or promote products derived from this software

without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## 1.15 zlib 1.2.12

### 1.15.1 Available under license :

\* zlib.h -- interface of the 'zlib' general purpose compression library  
version 1.2.12, March 11th, 2022

Copyright (C) 1995-2022 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly      Mark Adler  
jloup@gzip.org      madler@alumni.caltech.edu

The data format used by the zlib library is described by RFCs (Request for Comments) 1950 to 1952 in the files <http://tools.ietf.org/html/rfc1950> (zlib format), rfc1951 (deflate format) and rfc1952 (gzip format).

\*/

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## 1.16 appcompat 1.6.0

### 1.16.1 Available under license :

No license file was found, but licenses were detected in source scan.

```
/*
 * Copyright (C) 2022 AlexMofer
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
*/
```

Found in path(s):

```
* /opt/cola/permits/1549830853_1675235102.9042222/0/appcompat-1-6-0-sources-
```

jar/com/am/appcompat/view/ToggleView.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/dialog/MaterialAlertDialog.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/view/FloatingActionButton.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/view/OverflowListView.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/dialog/MaterialDialogFragment.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/widget/ShapeableImageView.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/app/PendingIntentCompat.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/view/LayoutParams.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/dialog/MaterialDialog.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/app/ToolbarDelegate.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/storage/StorageManagerCompat.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/storage/StorageVolumeCompat.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/view/ActionModeCompat.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/graphics/CanvasCompat.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/os/BundleCompat.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/widget/AppCompatImageView.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/dialog/MaterialDialogHelper.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/widget/DividerItemDecoration.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/view/MenuListView.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/view/MainLayout.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/dialog/MaterialAlertDialogFragment.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/view/MenuItemView.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/widget/ImageViewHelper.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/app/BackPressedDelegate.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-

jar/com/am/appcompat/app/ApplicationCompat.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/widget/AppCompatImageButton.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/view/ToggleLayout.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/view/FloatingPopupWindow.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/view/BoundsAdapter.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/view/BackgroundLayout.java  
No license file was found, but licenses were detected in source scan.

```
/*  
* Copyright (C) 2020 AlexMofer  
*  
* Licensed under the Apache License, Version 2.0 (the "License");  
* you may not use this file except in compliance with the License.  
* You may obtain a copy of the License at  
*  
* http://www.apache.org/licenses/LICENSE-2.0  
*  
* Unless required by applicable law or agreed to in writing, software  
* distributed under the License is distributed on an "AS IS" BASIS,  
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
* See the License for the specific language governing permissions and  
* limitations under the License.  
*/
```

Found in path(s):

\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/app/Fragment.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/app/ApplicationHolder.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/app/AppCompatDialogFragment.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/app/AppCompatAlertDialogFragment.java  
\* /opt/cola/permits/1549830853\_1675235102.9042222/0/appcompat-1-6-0-sources-  
jar/com/am/appcompat/app/AppCompatActivity.java

## 1.17openh264 3.35.0

### 1.17.1 Available under license :

Copyright (c) 2013, Cisco Systems  
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

\* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

\* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

# Contributors to the OpenH264 project

Patrick Ai  
Sijia Chen  
ZhaoZheng Chu  
Paley Du  
Martin Ettl  
Andreas Gal  
Xu Guang  
Licai Guo  
Yi Guo  
Horace Huang  
Steven Huang  
Ethan Hugg  
Cullen Jennings  
Zhaofeng Jia  
Derrick Jin  
Jesse Li  
Jifei Li  
Kai Li  
Karina Li  
Matt Li  
Xiang Li  
Bourne Ling  
Alex Liu  
Wayne Liu  
Varun Patil  
Eric Rescorla

Adam Roach  
Sawyer Shan  
Siping Tao  
Martin Storsj  
Brion Vibber  
James Wang  
Juanny Wang  
Zhiliang Wang  
Herv Willems  
Gregory J Wolfe  
Katherine Wu  
Guang Xu  
Jeffery Xu  
Gang Yang  
Li Yao  
Jiessie Zhang  
Rory Zhang  
Volvret Zhang  
Ling Zhu  
James Zhu  
Dong Zhang  
Haibo Zhu  
Huade Shi

## 1.18 libcxx 9.0.9svn

### 1.18.1 Available under license :

=====  
libc++ License  
=====

The libc++ library is dual licensed under both the University of Illinois "BSD-Like" license and the MIT license. As a user of this code you may choose to use it under either license. As a contributor, you agree to allow your code to be used under both.

Full text of the relevant licenses is included below.

=====  
University of Illinois/NCSA  
Open Source License

Copyright (c) 2009-2017 by the contributors listed in CREDITS.TXT

All rights reserved.

Developed by:

LLVM Team

University of Illinois at Urbana-Champaign

<http://llvm.org>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal with the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

- \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimers.
- \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimers in the documentation and/or other materials provided with the distribution.
- \* Neither the names of the LLVM Team, University of Illinois at Urbana-Champaign, nor the names of its contributors may be used to endorse or promote products derived from this Software without specific prior written permission.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE CONTRIBUTORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS WITH THE SOFTWARE.

=====  
Copyright (c) 2009-2014 by the contributors listed in CREDITS.TXT

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# People who have agreed to one of the CLAs and can contribute patches.

# The AUTHORS file lists the copyright holders; this file  
# lists people. For example, Google employees are listed here  
# but not in AUTHORS, because Google holds the copyright.  
#

# Names should be added to this file only after verifying that  
# the individual or the individual's organization has agreed to  
# the appropriate Contributor License Agreement, found here:  
#

# <https://developers.google.com/open-source/cla/individual>  
# <https://developers.google.com/open-source/cla/corporate>  
#

# The agreement for individuals can be filled out on the web.  
#

# When adding J Random Contributor's name to this file,  
# either J's name or J's organization's name should be  
# added to the AUTHORS file, depending on whether the  
# individual or corporate CLA was used.  
#

# Names should be added to this file as:

# Name <email address>

#

# Please keep the list sorted.

Albert Pretorius <pretoalb@gmail.com>

Arne Beer <arne@twobeer.de>

Billy Robert O'Neal III <billy.oneal@gmail.com> <bion@microsoft.com>

Chris Kennelly <ckennelly@google.com> <ckennelly@ckennelly.com>

Christopher Seymour <chris.j.seymour@hotmail.com>

Cyrille Faucheux <cyrille.faucheux@gmail.com>

David Coeurjolly <david.coeurjolly@liris.cnrs.fr>

Deniz Evrenci <denizevrenci@gmail.com>

Dominic Hamon <dma@stripsock.com> <dominic@google.com>

Dominik Czarnota <dominik.b.czarnota@gmail.com>

Eric Fiselier <eric@efcs.ca>

Eugene Zhuk <eugene.zhuk@gmail.com>

Evgeny Safronov <division494@gmail.com>

Federico Ficarelli <federico.ficarelli@gmail.com>

Felix Homann <linuxaudio@showlabor.de>

Ismael Jimenez Martinez <ismael.jimenez.martinez@gmail.com>

Jern-Kuan Leong <jernkuan@gmail.com>  
JianXiong Zhou <zhoujianxiong2@gmail.com>  
Joao Paulo Magalhaes <joaoppmagalhaes@gmail.com>  
John Millikin <jmillikin@stripe.com>  
Jussi Knuuttila <jussi.knuuttila@gmail.com>  
Kai Wolf <kai.wolf@gmail.com>  
Kishan Kumar <kumar.kishan@outlook.com>  
Kaito Udagawa <umireon@gmail.com>  
Lei Xu <eddyxu@gmail.com>  
Matt Clarkson <mattyclarkson@gmail.com>  
Maxim Vafin <maxvafin@gmail.com>  
Nick Hutchinson <nshutchinson@gmail.com>  
Oleksandr Sochka <sasha.sochka@gmail.com>  
Ori Livneh <ori.livneh@gmail.com>  
Pascal Leroy <phl@google.com>  
Paul Redmond <paul.redmond@gmail.com>  
Pierre Phaneuf <pphaneuf@google.com>  
Radoslav Yovchev <radoslav.tm@gmail.com>  
Raul Marin <rmrodriguez@cartodb.com>  
Ray Glover <ray.glover@uk.ibm.com>  
Robert Guo <robert.guo@mongodb.com>  
Roman Lebedev <lebedev.ri@gmail.com>  
Shuo Chen <chenshuo@chenshuo.com>  
Tobias Ulvgrd <tobias.ulvgard@dirac.se>  
Tom Madams <tom.ej.madams@gmail.com> <tmadams@google.com>  
Yixuan Qiu <yixuanq@gmail.com>  
Yusuke Suzuki <utatane.tea@gmail.com>  
Zbigniew Skowron <zbychs@gmail.com>

Apache License  
Version 2.0, January 2004  
<http://www.apache.org/licenses/>

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or

otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual,

worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents

of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.

Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

## 1.19 lifecycle-extensions 2.5.1

### 1.19.1 Available under license :

List of 3rd party licenses:

-----  
\* javapoet.jar (com.squareup:javapoet:1.8.0)

\*\*\*\*\* LICENSE:

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
  - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

- (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. Disclaimer of Warranty. Unless required by applicable law or

agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[ ]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

-----  
\* kotlin-stdlib.jar (org.jetbrains.kotlin:kotlin-stdlib:1.1.1)

\*\*\*\*\* NOTICE:

=====  
== NOTICE file corresponding to the section 4 d of ==  
== the Apache License, Version 2.0, ==  
== in this case for the Kotlin Compiler distribution. ==  
=====

Kotlin Compiler

Copyright 2010-2015 JetBrains s.r.o and respective authors and developers

\*\*\*\*\* LICENSE:

/\*

\* Copyright 2010-2017 JetBrains s.r.o.

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

\*\*\*\*\* LICENSE:

Apache License  
Version 2.0, January 2004

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally

submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
  - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
  - (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. **Submission of Contributions.** Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
6. **Trademarks.** This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. **Disclaimer of Warranty.** Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or

implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
  
9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

## END OF TERMS AND CONDITIONS

### APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

-----  
\* auto-common.jar (com.google.auto:auto-common:0.6)

\*\*\*\*\* LICENSE:

Apache License  
Version 2.0, January 2004  
<http://www.apache.org/licenses/>

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation,

and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s)

with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
- (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. **Submission of Contributions.** Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.  
Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
6. **Trademarks.** This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. **Disclaimer of Warranty.** Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. **Limitation of Liability.** In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
9. **Accepting Warranty or Additional Liability.** While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[ ]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

-----  
\* annotations.jar (org.jetbrains:annotations:13.0)

\*\*\*\*\* LICENSE:

Apache License  
Version 2.0, January 2004  
<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity

on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
  - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
  - (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
  - (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
  - (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one

of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. **Submission of Contributions.** Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.

Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. **Trademarks.** This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. **Disclaimer of Warranty.** Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. **Limitation of Liability.** In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a

result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. **Accepting Warranty or Additional Liability.** While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

## END OF TERMS AND CONDITIONS

### APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[ ]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

-----

\* guava.jar (com.google.guava:guava:18.0)

\*\*\*\*\* LICENSE:

Apache License  
Version 2.0, January 2004  
<http://www.apache.org/licenses/>

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain

separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
- (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.

Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the

origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

## END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[ ]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.

/\*

\* Copyright (C) 2017 The Android Open Source Project

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

## 1.20 guava 30.1.1-android

### 1.20.1 Available under license :

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2015 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/package-info.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2007 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express

\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.

\*/

/\*

\* This following method is a modified version of one found in

\* <http://gee.cs.oswego.edu/cgi-bin/viewcvs.cgi/jsr166/src/test/tck/AbstractExecutorServiceTest.java?revision=1.30>

\* which contained the following notice:

\*

\* Written by Doug Lea with assistance from members of JCP JSR-166 Expert Group and released to  
\* the public domain, as explained at <http://creativecommons.org/publicdomain/zero/1.0/>

\*

\* Other contributors include Andrew Wright, Jeffrey Hayes, Pat Fisher, Mike Judd.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/MoreExecutors.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2010 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either

express

\* or implied. See the License for the specific language governing permissions and limitations under

\* the License.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/net/package-info.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/Atomics.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Strings.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/ThreadFactoryBuilder.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Ascii.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ContiguousSet.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/Monitor.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/ForwardingBlockingQueue.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Equivalence.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/SortedLists.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/UncaughtExceptionHandler.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/package-info.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/annotations/Beta.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/ListeningExecutorService.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/annotations/package-info.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2019 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/IncidentEdgeSet.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/CompactHashing.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2014 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express

\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/eventbus/SubscriberRegistry.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/MoreObjects.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/math/Quantiles.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/eventbus/Subscriber.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/eventbus/Dispatcher.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/TrustedListenableFutureTask.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/ListenerCallQueue.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2009 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express  
\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/AbstractExecutionThreadService.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/ForwardingFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/Splitter.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/ByteArrayDataInput.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/annotations/GwtIncompatible.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/ForwardingListenableFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/MapMakerInternalMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/Service.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/annotations/GwtCompatible.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/escape/ArrayBasedUnicodeEscaper.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Cut.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/escape/Escapers.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/net/HostSpecifier.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/cache/LocalCache.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/LineProcessor.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/JdkFutureAdapters.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/AbstractIdleService.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/io/ByteArrayDataOutput.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/cache/CacheBuilder.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/AbstractService.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/SparseImmutableTable.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/RegularImmutableTable.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/TypeResolver.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/MapMaker.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/cache/ReferenceEntry.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/net/InternetDomainName.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/html/HtmlEscapers.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/DenseImmutableTable.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/net/UrlEscapers.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/Callables.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/escape/ArrayBasedEscaperMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/SettableFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/primitives/SignedBytes.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/ForwardingFluentFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/escape/ArrayBasedCharEscaper.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/ByteProcessor.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/xml/XmlEscapers.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/primitives/UnsignedBytes.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/escape/Platform.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/Platform.java

No license file was found, but licenses were detected in source scan.

/\*

```
* Copyright (C) 2008 The Guava Authors
*
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
*/
/*
* This method was rewritten in Java from an intermediate step of the Murmur hash function in
* http://code.google.com/p/smhasher/source/browse/trunk/MurmurHash3.cpp, which contained the
* following header:
*
* MurmurHash3 was written by Austin Appleby, and is placed in the public domain. The author
* hereby disclaims copyright to this source code.
*/
```

Found in path(s):

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/collect/Hashing.java
```

No license file was found, but licenses were detected in source scan.

```
/*
* Copyright (C) 2020 The Guava Authors
*
* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except
* in compliance with the License. You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software distributed under the License
* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
express
* or implied. See the License for the specific language governing permissions and limitations under
* the License.
*/
/**
* Holder for web specializations of methods of { @code Floats }. Intended to be empty for regular
* version.
*/
```

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/FloatsMethodsForWeb.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2011 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express

\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.

\*/

/\*

\* This method was written by Doug Lea with assistance from members of JCP JSR-166 Expert Group

\* and released to the public domain, as explained at

\* <http://creativecommons.org/licenses/publicdomain>

\*

\* As of 2010/06/11, this method is identical to the (package private) hash method in OpenJDK 7's

\* `java.util.HashMap` class.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/Striped.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2011 The Guava Authors.

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express

\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/package-info.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2013 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except

\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License

\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express

\* or implied. See the License for the specific language governing permissions and limitations under

\* the License.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/AbstractTable.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/reflect/TypeVisitor.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/VerifyException.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/Runnables.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Verify.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/HashingInputStream.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/WrappingScheduledExecutorService.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/eventbus/SubscriberExceptionContext.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/thirdparty/publicsuffix/PublicSuffixType.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Utf8.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/FilteredMultimapValues.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/io/CharSequenceReader.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/eventbus/SubscriberExceptionHandler.java

No license file was found, but licenses were detected in source scan.

```
/*
 * Copyright (C) 2013 The Guava Authors
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
```

Found in path(s):

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/collect/MultimapBuilder.java
```

No license file was found, but licenses were detected in source scan.

```
/*
 * Copyright (C) 2007 The Guava Authors
 *
 * Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except
 * in compliance with the License. You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software distributed under the License
 * is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express
 * or implied. See the License for the specific language governing permissions and limitations under
 * the License.
 */
```

Found in path(s):

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/base/Preconditions.java
```

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/io/Resources.java
```

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/io/package-info.java
```

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/base/FinalizablePhantomReference.java
```

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/base/Suppliers.java
```

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
```

jar/com/google/common/base/Charsets.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/LineReader.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/FinalizableSoftReference.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/CharStreams.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/Predicate.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/ByteStreams.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/primitives/Primitives.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/LineBuffer.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/FinalizableWeakReference.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/Predicates.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/Functions.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/AbstractIterator.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/package-info.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/HashBiMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/eventbus/DeadEvent.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/FinalizableReference.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/eventbus/Subscribe.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/Objects.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/Supplier.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/MultiInputStream.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Interners.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/EnumMultiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/ExecutionList.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/package-info.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/base/Defaults.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/Function.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/FinalizableReferenceQueue.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/AbstractFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/Closeables.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/eventbus/AllowConcurrentEvents.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/Files.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/eventbus/AsyncEventBus.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/LittleEndianDataOutputStream.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/eventbus/EventBus.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/CountingOutputStream.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/Throwables.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/ListenableFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/DirectExecutor.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/CountingInputStream.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/LittleEndianDataInputStream.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/Flushables.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/eventbus/package-info.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2008 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except

\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License

\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express

\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Converter.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/net/PercentEscaper.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/thirdparty/publicsuffix/TrieParser.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/Booleans.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/internal/Finalizer.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/Doubles.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/CharMatcher.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/Floats.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/SequentialExecutor.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/Bytes.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/ListenableFutureTask.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/FluentIterable.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Stopwatch.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/io/FileBackedOutputStream.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/Shorts.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/Ints.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/Chars.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/io/MultiReader.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Joiner.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/escape/UnicodeEscaper.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/Longs.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/net/InetAddresses.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/escape/Escaper.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2011 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except

\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the

\* License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND,

either

\* express or implied. See the License for the specific language governing permissions and

\* limitations under the License.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/ImmutableSortedMultisetFauxverideShim.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/RangeSet.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/SortedIterable.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/SortedIterables.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/AbstractRangeSet.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/ImmutableSortedMultiset.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/GeneralRange.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/RegularImmutableSortedMultiset.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/ForwardingSortedMultiset.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/Count.java

No license file was found, but licenses were detected in source scan.

/\*

\* Written by Doug Lea with assistance from members of JCP JSR-166

\* Expert Group and released to the public domain, as explained at

\* <http://creativecommons.org/publicdomain/zero/1.0/>

\*/

Found in path(s):

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/hash/LongAdder.java  
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/cache/Striped64.java  
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/cache/LongAdder.java  
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/hash/Striped64.java  
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/AtomicDoubleArray.java
```

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2020 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express

\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.

\*/

/\*\*

\* Holder for web specializations of methods of { @code Doubles }. Intended to be empty for regular  
\* version.

\*/

Found in path(s):

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/primitives/DoublesMethodsForWeb.java
```

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2009 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

```
* distributed under the License is distributed on an "AS IS" BASIS,  
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
* See the License for the specific language governing permissions and  
* limitations under the License.  
*/  
/**  
 * Not supported. <b>You are attempting to create a map that may contain a non-{@code Comparable}  
 * key.</b> Proper calls will resolve to the version in {@code ImmutableSortedMap}, not this dummy  
 * version.  
 *  
 * @throws UnsupportedOperationException always  
 * @deprecated <b>Pass a key of type {@code Comparable} to use {@link  
 *   ImmutableSortedMap#of(Comparable, Object)}.</b>  
 */
```

Found in path(s):

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ImmutableSortedMapFauxverideShim.java  
No license file was found, but licenses were detected in source scan.
```

```
/*  
 * Copyright (C) 2005 The Guava Authors  
 *  
 * Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
 * in compliance with the License. You may obtain a copy of the License at  
 *  
 * http://www.apache.org/licenses/LICENSE-2.0  
 *  
 * Unless required by applicable law or agreed to in writing, software distributed under the License  
 * is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
 * express  
 * or implied. See the License for the specific language governing permissions and limitations under  
 * the License.  
 */
```

Found in path(s):

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/Reflection.java  
No license file was found, but licenses were detected in source scan.
```

```
/*  
 * Copyright (C) 2011 The Guava Authors  
 *  
 * Licensed under the Apache License, Version 2.0 (the "License");  
 * you may not use this file except in compliance with the License.  
 * You may obtain a copy of the License at  
 *  
 * http://www.apache.org/licenses/LICENSE-2.0
```

\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/GwtTransient.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright 2019 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express

\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/IgnoreJRERRequirement.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2016 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express

\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.

\*/

/\*\*

\* Holder for extra methods of {@code Objects} only in web. Intended to be empty for regular  
\* version.  
\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/ExtraObjectsMethodsForWeb.java  
No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2007 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

/\*\*

\* Returns an array containing all of the elements in the specified collection. This method

\* returns the elements in the order they are returned by the collection's iterator. The returned

\* array is "safe" in that no references to it are maintained by the collection. The caller is

\* thus free to modify the returned array.

\*

\* <p>This method assumes that the collection size doesn't change while the method is running.

\*

\* <p>TODO(kevinb): support concurrently modified collections?

\*

\* @param c the collection for which to return an array of elements

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ObjectArrays.java  
No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2014 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/TopKSelector.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/SuccessorsFunction.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/Graph.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/PredecessorsFunction.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/ImmutableGraph.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/Graphs.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/MutableGraph.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/Network.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/MutableNetwork.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/ImmutableNetwork.java

No license file was found, but licenses were detected in source scan.

/\*  
\* Copyright (C) 2012 The Guava Authors  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.  
\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/FilteredEntrySetMultimap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ForwardingImmutableSet.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/SortedMultisetBridge.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/DescendingImmutableSortedSet.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/FilteredKeySetMultimap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableEnumMap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/FilteredEntryMultimap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/DescendingMultiset.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/TreeRangeMap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ForwardingNavigableSet.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/AllEqualOrdering.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/TreeTraverser.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/RegularImmutableAsList.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/CompactHashMap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/TransformedIterator.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/ForwardingBlockingDeque.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ForwardingDeque.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/AbstractSortedKeySortedSetMultimap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ForwardingNavigableMap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/TransformedListIterator.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ForwardingBlockingDeque.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ForwardingImmutableList.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/AbstractMultimap.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/CompactHashSet.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/EvictingQueue.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/FilteredSetMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/CompactLinkedHashMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/CompactLinkedHashSet.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/AbstractNavigableMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ForwardingImmutableMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/FilteredKeyListMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/UnmodifiableSortedMultiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/RangeMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/FilteredMultimap.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2016 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express

\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/JdkPattern.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/PatternCompiler.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/CommonPattern.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/CommonMatcher.java

No license file was found, but licenses were detected in source scan.

```
/*
 * Copyright (C) 2015 The Guava Authors
 *
 * Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except
 * in compliance with the License. You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software distributed under the License
 * is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express
 * or implied. See the License for the specific language governing permissions and limitations under
 * the License.
 */
```

Found in path(s):

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/hash/LittleEndianByteArray.java
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/util/concurrent/InterruptibleTask.java
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/hash/MacHashFunction.java
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/io/ReaderInputStream.java
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/util/concurrent/AggregateFutureState.java
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/util/concurrent/Platform.java
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/collect/ConsumingQueueIterator.java
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/util/concurrent/AsyncCallable.java
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/hash/FarmHashFingerprint64.java
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/util/concurrent/CombinedFuture.java
```

No license file was found, but licenses were detected in source scan.

```
/*
 * Copyright (C) 2018 The Guava Authors
 *
 * Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except
 * in compliance with the License. You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
```

\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express  
\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/hash/ImmutableSupplier.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/ExecutionSequencer.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2020 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express  
\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/math/BigDecimalMath.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/Java8Compatibility.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/OverflowAvoidingLockSupport.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/hash/Java8Compatibility.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/math/ToDoubleRounder.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2009 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at

\*

```
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software distributed under the License
* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
express
* or implied. See the License for the specific language governing permissions and limitations under
* the License.
*/
/**
* Outer class that exists solely to let us write {@code Partially.GwtIncompatible} instead of plain
* {@code GwtIncompatible}. This is more accurate for {@link Futures#catching}, which is available
* under GWT but with a slightly different signature.
*
* <p>We can't use {@code PartiallyGwtIncompatible} because then the GWT compiler wouldn't recognize
* it as a {@code GwtIncompatible} annotation. And for {@code Futures.catching}, we need the GWT
* compiler to autostrip the normal server method in order to expose the special, inherited GWT
* version.
*/
```

Found in path(s):

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/util/concurrent/Partially.java
No license file was found, but licenses were detected in source scan.
```

```
/*
* Copyright (C) 2020 The Guava Authors
*
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
*/
```

Found in path(s):

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/util/concurrent/ServiceManagerBridge.java
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/base/Java8Usage.java
No license file was found, but licenses were detected in source scan.
```

```
/*
```

\* Copyright (C) 2012 The Guava Authors  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express  
\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/ImmutableTypeToInstanceMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/MutableTypeToInstanceMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/CharSource.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/CartesianList.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ImmutableRangeMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/escape/package-info.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/hash/LongAddable.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/BaseEncoding.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/ByteSource.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/math/PairedStatsAccumulator.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/AbstractInvocationHandler.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/ClassPath.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/math/StatsAccumulator.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/Closer.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/xml/package-info.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/SmoothRateLimiter.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/util/concurrent/ListenableScheduledFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/math/PairedStats.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/Invokable.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/TypeCapture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/math/Stats.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/RateLimiter.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/Parameter.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/TypeToInstanceMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/Element.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/cache/LongAddables.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/FileWriteMode.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/hash/LongAddables.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/ByteSink.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/ServiceManager.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/base/StandardSystemProperty.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/CharSink.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/hash/AbstractByteHasher.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/math/LinearTransformation.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/html/package-info.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/hash/SipHashFunction.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/package-info.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/hash/ChecksumHashFunction.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ImmutableRangeSet.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/FilteredKeyMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/cache/LongAddable.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2017 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except

\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License

\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express

\* or implied. See the License for the specific language governing permissions and limitations under

\* the License.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/hash/AbstractHashFunction.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/ForwardingLock.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/ForwardingCondition.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/primitives/ImmutableDoubleArray.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/primitives/ImmutableIntArray.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/primitives/ImmutableLongArray.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2017 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

Found in path(s):

- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ObjectCountLinkedHashMap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/Traverser.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/AbstractBaseGraph.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/BaseGraph.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ObjectCountHashMap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/ClosingFuture.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2011 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express

\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.

\*/

Found in path(s):

- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/package-info.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/AbstractHasher.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/AbstractListeningExecutorService.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/UnsignedInts.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/UnsignedLongs.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/ListeningScheduledExecutorService.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/Crc32cHashFunction.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/CacheLoader.java

- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/AbstractCache.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/math/package-info.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/AsyncFunction.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/AbstractNonStreamingHashFunction.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/ParseRequest.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/Funnel.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/RemovalNotification.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/TreeRangeSet.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/Weigher.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/Uninterruptibles.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/net/HostAndPort.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/AbstractCompositeHashFunction.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/BloomFilterStrategies.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/HashCode.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/ExecutionError.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/MessageDigestHashFunction.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/Murmur3\_128HashFunction.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/math/IntMath.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/Hashing.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Optional.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/UncheckedExecutionException.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/math/LongMath.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/BoundType.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/UnsignedLong.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/BloomFilter.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/CycleDetectingLockFactory.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/RemovalListeners.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/Queues.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/math/BigIntegerMath.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/CacheBuilderSpec.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/net/HttpHeaders.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/ForwardingExecutorService.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/DescendingImmutableSortedMultiset.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Ticker.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/HashingOutputStream.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/reflect/Types.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/Funnels.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/FutureCallback.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/ForwardingLoadingCache.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/PrimitiveSink.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/HashFunction.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/math/DoubleMath.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/RemovalListener.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/Hasher.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Enums.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/LoadingCache.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Present.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/math/MathPreconditions.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/AbstractSortedMultiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/AbstractScheduledService.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/RegularContiguousSet.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/CacheStats.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/AbstractStreamingHasher.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/Cache.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/AtomicLongMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/RemovalCause.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/WrappingExecutorService.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/PairwiseEquivalence.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/ForwardingListeningExecutorService.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/AbstractLoadingCache.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/Absent.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/cache/ForwardingCache.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/FunctionalEquivalence.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/hash/Murmur3\_32HashFunction.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/math/DoubleUtils.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/EmptyContiguousSet.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/reflect/TypeParameter.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/RegularImmutableMultiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/primitives/UnsignedInteger.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/net/MediaType.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2011 The Guava Authors

\*  
\* Licensed under the Apache License, Version 2.0 (the "License"); you may not  
\* use this file except in compliance with the License. You may obtain a copy of  
\* the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS, WITHOUT  
\* WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the  
\* License for the specific language governing permissions and limitations under  
\* the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/SortedMultiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/SortedMultisets.java

No license file was found, but licenses were detected in source scan.

/\*  
\* Copyright (C) 2016 The Guava Authors  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.  
\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/AbstractNetwork.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/UndirectedMultiNetworkConnections.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/LinkedHashMultimapGwtSerializationDependencies.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/UndirectedNetworkConnections.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/graph/DirectedMultiNetworkConnections.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/MapIteratorCache.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/ForwardingNetwork.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/DirectedGraphConnections.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/StandardValueGraph.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/HashMultimapGwtSerializationDependencies.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/NetworkBuilder.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/EdgesConnecting.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableMultisetGwtSerializationDependencies.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/ElementOrder.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/StandardMutableGraph.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/GraphConstants.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/ImmutableValueGraph.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/AbstractGraphBuilder.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/NetworkConnections.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/MutableValueGraph.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/StandardMutableValueGraph.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/AbstractGraph.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/StandardNetwork.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/MapRetrievalCache.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/AbstractDirectedNetworkConnections.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/UndirectedGraphConnections.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/ValueGraph.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/GraphBuilder.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/EndpointPairIterator.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/Comparators.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/DirectedNetworkConnections.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/ForwardingGraph.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/StandardMutableNetwork.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/GraphConnections.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/EndpointPair.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/RangeGwtSerializationDependencies.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ArrayListMultimapGwtSerializationDependencies.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/MultiEdgesConnecting.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/AbstractUndirectedNetworkConnections.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/ForwardingValueGraph.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/AbstractValueGraph.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/graph/ValueGraphBuilder.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2010 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ForwardingSetMultimap.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/ForwardingSortedSetMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/RowSortedTable.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingImmutableCollection.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/AbstractSequentialIterator.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/UnmodifiableListIterator.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingListMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/MinMaxPriorityQueue.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/SortedMapDifference.java  
No license file was found, but licenses were detected in source scan.

```
/*  
* Copyright (C) 2019 The Guava Authors  
*  
* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
* in compliance with the License. You may obtain a copy of the License at  
*  
* http://www.apache.org/licenses/LICENSE-2.0  
*  
* Unless required by applicable law or agreed to in writing, software distributed under the License  
* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express  
* or implied. See the License for the specific language governing permissions and limitations under  
* the License.  
*/
```

Found in path(s):  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/primitives/Platform.java  
No license file was found, but licenses were detected in source scan.

```
/*  
* Copyright (C) 2009 The Guava Authors  
*  
* Licensed under the Apache License, Version 2.0 (the "License");  
* you may not use this file except in compliance with the License.  
* You may obtain a copy of the License at  
*  
* http://www.apache.org/licenses/LICENSE-2.0  
*  
* Unless required by applicable law or agreed to in writing, software  
* distributed under the License is distributed on an "AS IS" BASIS,
```

- \* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
- \* See the License for the specific language governing permissions and
- \* limitations under the License.
- \*/

Found in path(s):

- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableClassToInstanceMap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/RegularImmutableSortedSet.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ArrayTable.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableSortedMap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ForwardingTable.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/DiscreteDomain.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/RegularImmutableList.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableSetMultimap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableAsList.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/SingletonImmutableTable.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/EmptyImmutableSetMultimap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableSortedSetFauxverideShim.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ComputationException.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ComparisonChain.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/AbstractIndexedListIterator.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableEnumSet.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableTable.java

No license file was found, but licenses were detected in source scan.

/\*

- \* Copyright (C) 2020 The Guava Authors
- \*
- \* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except
- \* in compliance with the License. You may obtain a copy of the License at
- \*

```
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software distributed under the License
* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
express
* or implied. See the License for the specific language governing permissions and limitations under
* the License.
*/
/**
* Holder for web specializations of methods of { @code Ints }. Intended to be empty for regular
* version.
*/
```

Found in path(s):

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/primitives/IntsMethodsForWeb.java
```

No license file was found, but licenses were detected in source scan.

```
/*
* Copyright (C) 2012 The Guava Authors
*
* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except
* in compliance with the License. You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software distributed under the License
* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
express
* or implied. See the License for the specific language governing permissions and limitations under
* the License.
*/
/**
* This method was rewritten in Java from an intermediate step of the Murmur hash function in
* http://code.google.com/p/smhasher/source/browse/trunk/MurmurHash3.cpp, which contained the
* following header:
*
* MurmurHash3 was written by Austin Appleby, and is placed in the public domain. The author
* hereby disclaims copyright to this source code.
*/
```

Found in path(s):

```
* /opt/cola/permits/1148655839_1618904031.03/0/guava-30-1-1-android-sources-
jar/com/google/common/base/SmallCharMatcher.java
```

No license file was found, but licenses were detected in source scan.

```
/*
* Copyright (C) 2006 The Guava Authors
```

\*  
\* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
\* in compliance with the License. You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software distributed under the License  
\* is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
express  
\* or implied. See the License for the specific language governing permissions and limitations under  
\* the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/TimeoutFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/GwtFuturesCatchingSpecialization.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/annotations/VisibleForTesting.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/GwtFluentFutureCatchingSpecialization.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/FuturesGetChecked.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/SimpleTimeLimiter.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/FakeTimeLimiter.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/io/AppendableWriter.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/Futures.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/reflect/TypeToken.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/escape/CharEscaperBuilder.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/FluentFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/AbstractTransformFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/AbstractCatchingFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/AggregateFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/escape/CharEscaper.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/util/concurrent/CollectionFuture.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/ImmediateFuture.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/io/PatternFilenameFilter.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/TimeLimiter.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/util/concurrent/UncheckedTimeoutException.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/base/CaseFormat.java

No license file was found, but licenses were detected in source scan.

/\*

\* Copyright (C) 2008 The Guava Authors

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* <http://www.apache.org/licenses/LICENSE-2.0>

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableEntry.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/thirdparty/publicsuffix/PublicSuffixPatterns.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/CollectPreconditions.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableMapEntrySet.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableMapKeySet.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableCollection.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/Table.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableSortedSet.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/UnmodifiableIterator.java

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/ImmutableMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ImmutableBiMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/TreeBasedTable.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/RegularImmutableBiMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Platform.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ImmutableListMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Tables.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Range.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/HashBasedTable.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ImmutableMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/RegularImmutableMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Collections2.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ImmutableMultiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/StandardTable.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Serialization.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/StandardRowSortedTable.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/PeekingIterator.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/EmptyImmutableListMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ImmutableMapValues.java  
No license file was found, but licenses were detected in source scan.

/\*  
\* Copyright (C) 2018 The Guava Authors  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.  
\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\*

- \* Unless required by applicable law or agreed to in writing, software
- \* distributed under the License is distributed on an "AS IS" BASIS,
- \* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
- \* See the License for the specific language governing permissions and
- \* limitations under the License.

\*/

Found in path(s):

- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/IndexedImmutableSet.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/BaseImmutableMultimap.java

No license file was found, but licenses were detected in source scan.

/\*

- \* Copyright (C) 2007 The Guava Authors
- \*
- \* Licensed under the Apache License, Version 2.0 (the "License");
- \* you may not use this file except in compliance with the License.
- \* You may obtain a copy of the License at
- \*
- \* <http://www.apache.org/licenses/LICENSE-2.0>
- \*
- \* Unless required by applicable law or agreed to in writing, software
- \* distributed under the License is distributed on an "AS IS" BASIS,
- \* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
- \* See the License for the specific language governing permissions and
- \* limitations under the License.

\*/

Found in path(s):

- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/EnumBiMap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ForwardingSet.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ForwardingSortedSet.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ImmutableList.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/ForwardingCollection.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/AbstractSetMultimap.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/Multisets.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-jar/com/google/common/collect/Ordering.java
- \* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/NullsLastOrdering.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ImmutableSet.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingMapEntry.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ClassToInstanceMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/AbstractMapEntry.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/SingletonImmutableSet.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ListMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/SortedSetMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/TreeMultiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/LinkedListMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Interner.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/UsingToStringOrdering.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/AbstractIterator.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Maps.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingIterator.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ComparatorOrdering.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingSortedMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/SetMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Lists.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingMultiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Multiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/MutableClassToInstanceMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/RegularImmutableSet.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/package-info.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingQueue.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/AbstractBiMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/AbstractMapBasedMultiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ExplicitOrdering.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Iterators.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/LinkedHashMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ConcurrentHashMultiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Multimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/LexicographicalOrdering.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingConcurrentMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingObject.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/BiMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Iterables.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/AbstractSortedSetMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/MapDifference.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/AbstractListMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/HashMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Multimaps.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ReverseNaturalOrdering.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/AbstractMultiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/Sets.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingListIterator.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

jar/com/google/common/collect/Synchronized.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/CompoundOrdering.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/NaturalOrdering.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ReverseOrdering.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ArrayListMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ByFunctionOrdering.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/NullsFirstOrdering.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/ForwardingList.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/TreeMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/HashMultiset.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/AbstractMapBasedMultimap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/EnumHashBiMap.java  
\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-  
jar/com/google/common/collect/LinkedHashMultiset.java  
No license file was found, but licenses were detected in source scan.

```
/*  
 * Copyright (C) 2020 The Guava Authors  
 *  
 * Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except  
 * in compliance with the License. You may obtain a copy of the License at  
 *  
 * http://www.apache.org/licenses/LICENSE-2.0  
 *  
 * Unless required by applicable law or agreed to in writing, software distributed under the License  
 * is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either  
 * express  
 * or implied. See the License for the specific language governing permissions and limitations under  
 * the License.  
 */  
/**  
 * Holder for web specializations of methods of { @code Shorts}. Intended to be empty for regular  
 * version.  
 */
```

Found in path(s):

\* /opt/cola/permits/1148655839\_1618904031.03/0/guava-30-1-1-android-sources-

# 1.21 junit 4.13.2

## 1.21.1 Available under license :

JUnit

Eclipse Public License - v 1.0

THE ACCOMPANYING PROGRAM IS PROVIDED UNDER THE TERMS OF THIS ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THE PROGRAM CONSTITUTES RECIPIENT'S ACCEPTANCE OF THIS AGREEMENT.

### 1. DEFINITIONS

"Contribution" means:

- a) in the case of the initial Contributor, the initial code and documentation distributed under this Agreement, and
- b) in the case of each subsequent Contributor:
  - i) changes to the Program, and
  - ii) additions to the Program;

where such changes and/or additions to the Program originate from and are distributed by that particular Contributor. A Contribution 'originates' from a Contributor if it was added to the Program by such Contributor itself or anyone acting on such Contributor's behalf. Contributions do not include additions to the Program which: (i) are separate modules of software distributed in conjunction with the Program under their own license agreement, and (ii) are not derivative works of the Program.

"Contributor" means any person or entity that distributes the Program.

"Licensed Patents " mean patent claims licensable by a Contributor which are necessarily infringed by the use or sale of its Contribution alone or when combined with the Program.

"Program" means the Contributions distributed in accordance with this Agreement.

"Recipient" means anyone who receives the Program under this Agreement, including all Contributors.

### 2. GRANT OF RIGHTS

- a) Subject to the terms of this Agreement, each Contributor hereby grants

Recipient a non-exclusive, worldwide, royalty-free copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, distribute and sublicense the Contribution of such Contributor, if any, and such derivative works, in source code and object code form.

b) Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free patent license under Licensed Patents to make, use, sell, offer to sell, import and otherwise transfer the Contribution of such Contributor, if any, in source code and object code form. This patent license shall apply to the combination of the Contribution and the Program if, at the time the Contribution is added by the Contributor, such addition of the Contribution causes such combination to be covered by the Licensed Patents. The patent license shall not apply to any other combinations which include the Contribution. No hardware per se is licensed hereunder.

c) Recipient understands that although each Contributor grants the licenses to its Contributions set forth herein, no assurances are provided by any Contributor that the Program does not infringe the patent or other intellectual property rights of any other entity. Each Contributor disclaims any liability to Recipient for claims brought by any other entity based on infringement of intellectual property rights or otherwise. As a condition to exercising the rights and licenses granted hereunder, each Recipient hereby assumes sole responsibility to secure any other intellectual property rights needed, if any. For example, if a third party patent license is required to allow Recipient to distribute the Program, it is Recipient's responsibility to acquire that license before distributing the Program.

d) Each Contributor represents that to its knowledge it has sufficient copyright rights in its Contribution, if any, to grant the copyright license set forth in this Agreement.

### 3. REQUIREMENTS

A Contributor may choose to distribute the Program in object code form under its own license agreement, provided that:

a) it complies with the terms and conditions of this Agreement; and

b) its license agreement:

i) effectively disclaims on behalf of all Contributors all warranties and conditions, express and implied, including warranties or conditions of title and non-infringement, and implied warranties or conditions of merchantability and fitness for a particular purpose;

ii) effectively excludes on behalf of all Contributors all liability for damages, including direct, indirect, special, incidental and consequential

damages, such as lost profits;

iii) states that any provisions which differ from this Agreement are offered by that Contributor alone and not by any other party; and

iv) states that source code for the Program is available from such Contributor, and informs licensees how to obtain it in a reasonable manner on or through a medium customarily used for software exchange.

When the Program is made available in source code form:

a) it must be made available under this Agreement; and

b) a copy of this Agreement must be included with each copy of the Program.

Contributors may not remove or alter any copyright notices contained within the Program.

Each Contributor must identify itself as the originator of its Contribution, if any, in a manner that reasonably allows subsequent Recipients to identify the originator of the Contribution.

#### 4. COMMERCIAL DISTRIBUTION

Commercial distributors of software may accept certain responsibilities with respect to end users, business partners and the like. While this license is intended to facilitate the commercial use of the Program, the Contributor who includes the Program in a commercial product offering should do so in a manner which does not create potential liability for other Contributors. Therefore, if a Contributor includes the Program in a commercial product offering, such Contributor ("Commercial Contributor") hereby agrees to defend and indemnify every other Contributor ("Indemnified Contributor") against any losses, damages and costs (collectively "Losses") arising from claims, lawsuits and other legal actions brought by a third party against the Indemnified Contributor to the extent caused by the acts or omissions of such Commercial Contributor in connection with its distribution of the Program in a commercial product offering. The obligations in this section do not apply to any claims or Losses relating to any actual or alleged intellectual property infringement. In order to qualify, an Indemnified Contributor must: a) promptly notify the Commercial Contributor in writing of such claim, and b) allow the Commercial Contributor to control, and cooperate with the Commercial Contributor in, the defense and any related settlement negotiations. The Indemnified Contributor may participate in any such claim at its own expense.

For example, a Contributor might include the Program in a commercial product offering, Product X. That Contributor is then a Commercial Contributor. If that Commercial Contributor then makes performance claims, or offers warranties

related to Product X, those performance claims and warranties are such Commercial Contributor's responsibility alone. Under this section, the Commercial Contributor would have to defend claims against the other Contributors related to those performance claims and warranties, and if a court requires any other Contributor to pay any damages as a result, the Commercial Contributor must pay those damages.

## 5. NO WARRANTY

EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, THE PROGRAM IS PROVIDED ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OR CONDITIONS OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Each Recipient is solely responsible for determining the appropriateness of using and distributing the Program and assumes all risks associated with its exercise of rights under this Agreement, including but not limited to the risks and costs of program errors, compliance with applicable laws, damage to or loss of data, programs or equipment, and unavailability or interruption of operations.

## 6. DISCLAIMER OF LIABILITY

EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, NEITHER RECIPIENT NOR ANY CONTRIBUTORS SHALL HAVE ANY LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT LIMITATION LOST PROFITS), HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OR DISTRIBUTION OF THE PROGRAM OR THE EXERCISE OF ANY RIGHTS GRANTED HEREUNDER, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

## 7. GENERAL

If any provision of this Agreement is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this Agreement, and without further action by the parties hereto, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable.

If Recipient institutes patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Program itself (excluding combinations of the Program with other software or hardware) infringes such Recipient's patent(s), then such Recipient's rights granted under Section 2(b) shall terminate as of the date such litigation is filed.

All Recipient's rights under this Agreement shall terminate if it fails to comply with any of the material terms or conditions of this Agreement and does not cure such failure in a reasonable period of time after becoming aware of such noncompliance. If all Recipient's rights under this Agreement terminate,

Recipient agrees to cease use and distribution of the Program as soon as reasonably practicable. However, Recipient's obligations under this Agreement and any licenses granted by Recipient relating to the Program shall continue and survive.

Everyone is permitted to copy and distribute copies of this Agreement, but in order to avoid inconsistency the Agreement is copyrighted and may only be modified in the following manner. The Agreement Steward reserves the right to publish new versions (including revisions) of this Agreement from time to time. No one other than the Agreement Steward has the right to modify this Agreement. The Eclipse Foundation is the initial Agreement Steward. The Eclipse Foundation may assign the responsibility to serve as the Agreement Steward to a suitable separate entity. Each new version of the Agreement will be given a distinguishing version number. The Program (including Contributions) may always be distributed subject to the version of the Agreement under which it was received. In addition, after a new version of the Agreement is published, Contributor may elect to distribute the Program (including its Contributions) under the new version. Except as expressly stated in Sections 2(a) and 2(b) above, Recipient receives no rights or licenses to the intellectual property of any Contributor under this Agreement, whether expressly, by implication, estoppel or otherwise. All rights in the Program not expressly granted under this Agreement are reserved.

This Agreement is governed by the laws of the State of New York and the intellectual property laws of the United States of America. No party to this Agreement will bring a legal action under this Agreement more than one year after the cause of action arose. Each party waives its rights to a jury trial in any resulting litigation.

## 1.22 google-play-services 18.0.1

### 1.22.1 Available under license :

AndroidX lifecycle common library, AndroidX lifecycle livedatacore library, AndroidX lifecycle runtime library:

Copyright (c) 2005-2011, The Android Open Source Project

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
  - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

- (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. **Submission of Contributions.** Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
6. **Trademarks.** This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. **Disclaimer of Warranty.** Unless required by applicable law or

agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

apksig:

Copyright (c) 2005-2008, The Android Open Source Project

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- (a) You must give any other recipients of the Work or

Derivative Works a copy of this License; and

- (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.

Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
  
8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
  
9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

Animal Sniffer:

The MIT License

Copyright (c) 2008 Kohsuke Kawaguchi and codehaus.org.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is

furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Checker Framework Annotations:

A few parts of the Checker Framework have more permissive licenses.

\* The annotations are licensed under the MIT License. (The text of this license appears below.) More specifically, all the parts of the Checker Framework that you might want to include with your own program use the MIT License. This is the checker-qual.jar file and all the files that appear in it: every file in a qual/ directory, plus utility files such as NullnessUtil.java, RegexUtil.java, SignednessUtil.java, etc. In addition, the cleanroom implementations of third-party annotations, which the Checker Framework recognizes as aliases for its own annotations, are licensed under the MIT License.

=====

MIT License:

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN

Error Prone:

Apache License  
Version 2.0, January 2004  
<http://www.apache.org/licenses/>

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the

editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the

Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
- (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

## END OF TERMS AND CONDITIONS

### APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the

same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Google Auto, Guava JDK5, J2ObjC, JSR 250, SafeParcelable library:

Apache License  
Version 2.0, January 2004  
<http://www.apache.org/licenses/>

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation

source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable

(except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
- (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and

may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.

Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify,

defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

## END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Guava JDK7:

Apache License  
Version 2.0, January 2004  
<http://www.apache.org/licenses/>

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
  - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
  - (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
  - (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
  - (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not

pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.

Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special,

incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

## END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[ ]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

JSR 330:

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- (a) You must give any other recipients of the Work or

Derivative Works a copy of this License; and

- (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.

Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
  
8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
  
9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[ ]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.

JsInterop Annotations:

Apache License

Version 2.0, January 2004

<http://www.apache.org/licenses/>

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction,  
and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by  
the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all  
other entities that control, are controlled by, or are under common  
control with that entity. For the purposes of this definition,  
"control" means (i) the power, direct or indirect, to cause the  
direction or management of such entity, whether by contract or  
otherwise, or (ii) ownership of fifty percent (50%) or more of the  
outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity  
exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications,  
including but not limited to software source code, documentation  
source, and configuration files.

"Object" form shall mean any form resulting from mechanical  
transformation or translation of a Source form, including but  
not limited to compiled object code, generated documentation,  
and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You

institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
- (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. **Submission of Contributions.** Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.  
Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
6. **Trademarks.** This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. **Disclaimer of Warranty.** Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. **Limitation of Liability.** In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
9. **Accepting Warranty or Additional Liability.** While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

## APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[ ]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright 2017 Google Inc.

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

JSR 305:

Copyright (c) 2007-2009, JSR305 expert group  
All rights reserved.

<http://www.opensource.org/licenses/bsd-license.php>

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- \* Neither the name of the JSR305 expert group nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE

LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## 1.23 libxml2 2.9.10

### 1.23.1 Available under license :

Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar licence but with different Copyright notices) all the files are:

Copyright (C) 1998-2012 Daniel Veillard. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## 1.24 websocketpp 0.8.0

### 1.24.1 Available under license :

Main Library:

Copyright (c) 2014, Peter Thorson. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

- \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- \* Neither the name of the WebSocket++ Project nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL PETER THORSON BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Bundled Libraries:

\*\*\*\*\* Base 64 Library (base64/base64.hpp) \*\*\*\*\*

base64.hpp is a repackaging of the base64.cpp and base64.h files into a single header suitable for use as a header only library. This conversion was done by Peter Thorson (webmaster@zaphoyd.com) in 2012. All modifications to the code are redistributed under the same license as the original, which is listed below.

base64.cpp and base64.h

Copyright (C) 2004-2008 Ren Nyffenegger

This source code is provided 'as-is', without any express or implied warranty. In no event will the author be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this source code must not be misrepresented; you must not claim that you wrote the original source code. If you use this source code in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original source code.
3. This notice may not be removed or altered from any source distribution.

Ren Nyffenegger rene.nyffenegger@adp-gmbh.ch

\*\*\*\*\* SHA1 Library (sha1/sha1.hpp) \*\*\*\*\*

sha1.hpp is a repackaging of the sha1.cpp and sha1.h files from the shallsha1 library (<http://code.google.com/p/smallsha1/>) into a single header suitable for use as a header only library. This conversion was done by Peter Thorson (webmaster@zaphoyd.com) in 2013. All modifications to the code are redistributed under the same license as the original, which is listed below.

Copyright (c) 2011, Micael Hildenborg

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- \* Neither the name of Micael Hildenborg nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY Micael Hildenborg "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL Micael Hildenborg BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

\*\*\*\*\* MD5 Library (common/md5.hpp) \*\*\*\*\*

md5.hpp is a reformulation of the md5.h and md5.c code from <http://www.opensource.apple.com/source/cups/cups-59/cups/md5.c> to allow it to function as a component of a header only library. This conversion was done by Peter Thorson (webmaster@zaphoyd.com) in 2012 for the WebSocket++ project. The changes are released under the same license as the original (listed below)

Copyright (C) 1999, 2002 Aladdin Enterprises. All rights reserved.

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

L. Peter Deutsch  
ghost@aladdin.com

\*\*\*\*\* UTF8 Validation logic (utf8\_validation.hpp) \*\*\*\*\*

utf8\_validation.hpp is adapted from code originally written by Bjoern Hoehrmann <bjoern@hoehrmann.de>. See <http://bjoern.hoehrmann.de/utf-8/decoder/dfa/> for details.

The original license:

Copyright (c) 2008-2009 Bjoern Hoehrmann <bjoern@hoehrmann.de>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## 1.25 safestring 2.0

## 1.25.1 Available under license :

No license file was found, but licenses were detected in source scan.

```
/*-----  
* mem_primitives_lib.c - Unguarded Memory Copy Routines  
*  
* February 2005, Bo Berry  
*  
* Copyright (c) 2005-2010, 2013 by Cisco Systems, Inc.  
*  
* Permission is hereby granted, free of charge, to any person  
* obtaining a copy of this software and associated documentation  
* files (the "Software"), to deal in the Software without  
* restriction, including without limitation the rights to use,  
* copy, modify, merge, publish, distribute, sublicense, and/or  
* sell copies of the Software, and to permit persons to whom the  
* Software is furnished to do so, subject to the following  
* conditions:  
*  
* The above copyright notice and this permission notice shall be  
* included in all copies or substantial portions of the Software.  
*  
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,  
* EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES  
* OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND  
* NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT  
* HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY,  
* WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING  
* FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR  
* OTHER DEALINGS IN THE SOFTWARE.  
*-----  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-  
gz/safe_lib_2_0/src/safe_string/mem_primitives_lib.c
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* strcat_s.c  
*  
* October 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
* All rights Reserved.  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe_lib_2_0/src/safe_string/strcat_s.c
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* strprk_s.c  
*  
* November 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
* All rights Reserved.  
*-----  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe_lib_2_0/src/safe_string/strprk_s.c
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* strstr_s.c  
*  
* November 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
* All rights reserved.  
*-----  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe_lib_2_0/src/safe_string/strstr_s.c
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* safe_types.h -- types include for portability  
*  
* March 2007, Bo Berry  
*  
* Copyright (c) 2007-2010, 2013 by Cisco Systems, Inc.  
*  
* Permission is hereby granted, free of charge, to any person  
* obtaining a copy of this software and associated documentation  
* files (the "Software"), to deal in the Software without restriction,  
* including without limitation the rights to use, copy, modify, merge,  
* publish, distribute, sublicense, and/or sell copies of the Software,  
* and to permit persons to whom the Software is furnished to do so,  
* subject to the following conditions:  
*  
*-----
```

\* The above copyright notice and this permission notice shall be  
\* included in all copies or substantial portions of the Software.  
\*  
\* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,  
\* EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES  
\* OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.  
\* IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY  
\* CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,  
\* TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE  
\* SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
\*-----  
\*/

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe\_lib\_2\_0/include/safe\_types.h

No license file was found, but licenses were detected in source scan.

/\*-----  
\* safe\_str\_constraint.h  
\*  
\* October 2008, Bo Berry  
\*  
\* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
\*  
\* Permission is hereby granted, free of charge, to any person  
\* obtaining a copy of this software and associated documentation  
\* files (the "Software"), to deal in the Software without restriction,  
\* including without limitation the rights to use, copy, modify, merge,  
\* publish, distribute, sublicense, and/or sell copies of the Software,  
\* and to permit persons to whom the Software is furnished to do so,  
\* subject to the following conditions:  
\*  
\* The above copyright notice and this permission notice shall be  
\* included in all copies or substantial portions of the Software.  
\*  
\* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,  
\* EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES  
\* OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.  
\* IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY  
\* CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,  
\* TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE  
\* SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
\*-----  
\*/

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-

gz/safe\_lib\_2\_0/src/safe\_string/safe\_str\_constraint.h

No license file was found, but licenses were detected in source scan.

```
/*-----  
* safe_string.h  
*  
* November 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
*  
* Permission is hereby granted, free of charge, to any person  
* obtaining a copy of this software and associated documentation  
* files (the "Software"), to deal in the Software without restriction,  
* including without limitation the rights to use, copy, modify, merge,  
* publish, distribute, sublicense, and/or sell copies of the Software,  
* and to permit persons to whom the Software is furnished to do so,  
* subject to the following conditions:  
*  
* The above copyright notice and this permission notice shall be  
* included in all copies or substantial portions of the Software.  
*  
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,  
* EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES  
* OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.  
* IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY  
* CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,  
* TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE  
* SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
*-----  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe_lib_2_0/include/safe_string.h
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* strisuppercase_s.c  
*  
* October 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
* All rights reserved.  
*-----  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe_lib_2_0/src/safe_string/strisuppercase_s.c
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* mem_primitives_lib.h - Unguarded Memory Copy Routines  
*  
* October 2005, Bo Berry  
*  
* Copyright (c) 2005-2010, 2013 by Cisco Systems, Inc.  
*  
* Permission is hereby granted, free of charge, to any person  
* obtaining a copy of this software and associated documentation  
* files (the "Software"), to deal in the Software without restriction,  
* including without limitation the rights to use, copy, modify, merge,  
* publish, distribute, sublicense, and/or sell copies of the Software,  
* and to permit persons to whom the Software is furnished to do so,  
* subject to the following conditions:  
*  
* The above copyright notice and this permission notice shall be  
* included in all copies or substantial portions of the Software.  
*  
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,  
* EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES  
* OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.  
* IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY  
* CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,  
* TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE  
* SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
*-----  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-  
gz/safe_lib_2_0/include/mem_primitives_lib.h
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* strlastchar_s.c  
*  
* November 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
* All rights reserved  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-  
gz/safe_lib_2_0/src/safe_string/strlastchar_s.c
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* safe_mem_constraint.h  
*  
* October 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
*  
* Permission is hereby granted, free of charge, to any person  
* obtaining a copy of this software and associated documentation  
* files (the "Software"), to deal in the Software without restriction,  
* including without limitation the rights to use, copy, modify, merge,  
* publish, distribute, sublicense, and/or sell copies of the Software,  
* and to permit persons to whom the Software is furnished to do so,  
* subject to the following conditions:  
*  
* The above copyright notice and this permission notice shall be  
* included in all copies or substantial portions of the Software.  
*  
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,  
* EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES  
* OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.  
* IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY  
* CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,  
* TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE  
* SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
*-----  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-  
gz/safe_lib_2_0/src/safe_string/safe_mem_constraint.h
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* strzero_s.c  
*  
* October 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
* All Rights Reserved.  
*-----  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-
```

gz/safe\_lib\_2\_0/src/safe\_string/strzero\_s.c

No license file was found, but licenses were detected in source scan.

```
/*-----  
* safe_limits.h -- limits include for portability  
*  
* February 2009, Bo Berry  
*  
* Copyright (c) 2009-2010, 2013 by Cisco Systems, Inc.  
*  
* Permission is hereby granted, free of charge, to any person  
* obtaining a copy of this software and associated documentation  
* files (the "Software"), to deal in the Software without restriction,  
* including without limitation the rights to use, copy, modify, merge,  
* publish, distribute, sublicense, and/or sell copies of the Software,  
* and to permit persons to whom the Software is furnished to do so,  
* subject to the following conditions:  
*  
* The above copyright notice and this permission notice shall be  
* included in all copies or substantial portions of the Software.  
*  
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,  
* EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES  
* OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.  
* IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY  
* CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,  
* TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE  
* SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
*-----  
*/
```

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe\_lib\_2\_0/include/safe\_limits.h

No license file was found, but licenses were detected in source scan.

```
/*-----  
* strncat_s.c  
*  
* October 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
* All Rights reserved  
*-----  
*/
```

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-

gz/safe\_lib\_2\_0/src/safe\_string/strncat\_s.c

No license file was found, but licenses were detected in source scan.

```
/*-----  
* safe_lib.h -- library include for portability  
*  
* February 2009, Bo Berry  
*  
* Copyright (c) 2009-2010, 2013 by Cisco Systems, Inc.  
*  
* Permission is hereby granted, free of charge, to any person  
* obtaining a copy of this software and associated documentation  
* files (the "Software"), to deal in the Software without restriction,  
* including without limitation the rights to use, copy, modify, merge,  
* publish, distribute, sublicense, and/or sell copies of the Software,  
* and to permit persons to whom the Software is furnished to do so,  
* subject to the following conditions:  
*  
* The above copyright notice and this permission notice shall be  
* included in all copies or substantial portions of the Software.  
*  
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,  
* EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES  
* OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.  
* IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY  
* CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,  
* TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE  
* SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
*-----  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe_lib_2_0/include/safe_lib.h
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* strljustify_s.c  
*  
* November 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
* All rights Reserved  
*-----  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe_lib_2_0/src/safe_string/strljustify_s.c
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* safe_str_lib.h -- Safe C Library String APIs  
*  
* October 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
*  
* Permission is hereby granted, free of charge, to any person  
* obtaining a copy of this software and associated documentation  
* files (the "Software"), to deal in the Software without restriction,  
* including without limitation the rights to use, copy, modify, merge,  
* publish, distribute, sublicense, and/or sell copies of the Software,  
* and to permit persons to whom the Software is furnished to do so,  
* subject to the following conditions:  
*  
* The above copyright notice and this permission notice shall be  
* included in all copies or substantial portions of the Software.  
*  
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,  
* EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES  
* OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.  
* IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY  
* CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,  
* TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE  
* SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
*-----  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe_lib_2_0/include/safe_str_lib.h
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* strcpyfdin_s.c  
*  
* November 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
* All rights Reserved.  
*/
```

Found in path(s):

```
* /opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe_lib_2_0/src/safe_string/strcpyfdin_s.c
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* strncpy_s.c  
*  
* October 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
* All rights Reserved.  
*-----  
*/
```

Found in path(s):

```
*/opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe_lib_2_0/src/safe_string/strncpy_s.c
```

No license file was found, but licenses were detected in source scan.

```
/*-----  
* safe_mem_lib.h -- Safe C Library Memory APIs  
*  
* October 2008, Bo Berry  
*  
* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
*  
* Permission is hereby granted, free of charge, to any person  
* obtaining a copy of this software and associated documentation  
* files (the "Software"), to deal in the Software without restriction,  
* including without limitation the rights to use, copy, modify, merge,  
* publish, distribute, sublicense, and/or sell copies of the Software,  
* and to permit persons to whom the Software is furnished to do so,  
* subject to the following conditions:  
*  
* The above copyright notice and this permission notice shall be  
* included in all copies or substantial portions of the Software.  
*  
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,  
* EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES  
* OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.  
* IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY  
* CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,  
* TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE  
* SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.  
*-----  
*/
```

Found in path(s):

```
*/opt/cola/permits/1502360938_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe_lib_2_0/include/safe_mem_lib.h
```

No license file was found, but licenses were detected in source scan.

```
/*-----
```

\* strcasestr\_s.c  
\*  
\* November 2008, Bo Berry  
\*  
\* Copyright (c) 2008-2010, 2012-2013 by Cisco Systems, Inc.  
\* All rights Reserved.  
\*/

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe\_lib\_2\_0/src/safe\_string/strcasestr\_s.c

No license file was found, but licenses were detected in source scan.

/\*-----  
\* strspn\_s.c  
\*  
\* November 2008, Bo Berry  
\*  
\* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
\* All rights Reserved.  
\*-----  
\*/

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe\_lib\_2\_0/src/safe\_string/strspn\_s.c

No license file was found, but licenses were detected in source scan.

/\*-----  
\* strtok\_s.c  
\*  
\* October 2008, Bo Berry  
\*  
\* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.  
\* All rights reserved.  
\*-----  
\*/

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe\_lib\_2\_0/src/safe\_string/strtok\_s.c

No license file was found, but licenses were detected in source scan.

/\*-----  
\* strispassword\_s.c  
\*  
\* October 2008, Bo Berry  
\*  
\*/

\* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.

\* All rights Reserved.

\*-----

\*/

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe\_lib\_2\_0/src/safe\_string/strispassword\_s.c

No license file was found, but licenses were detected in source scan.

/\*-----

\* strislowercase\_s.c

\*

\* February 2005, Bo Berry

\*

\* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.

\* All rights Reserved.

\*-----

\*/

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe\_lib\_2\_0/src/safe\_string/strislowercase\_s.c

No license file was found, but licenses were detected in source scan.

/\*-----

\* stremovews\_s.c

\*

\* November 2008, Bo Berry

\*

\* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.

\* All rights reserved.

\*-----

\*/

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe\_lib\_2\_0/src/safe\_string/stremovews\_s.c

No license file was found, but licenses were detected in source scan.

/\*-----

\* strprefix\_s.c

\*

\* November 2008, Bo Berry

\*

\* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.

\* All rights reserved

\*-----

\*/

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe\_lib\_2\_0/src/safe\_string/strprefix\_s.c

No license file was found, but licenses were detected in source scan.

/\*-----

\* strlastsame\_s.c

\*

\* November 2008, Bo Berry

\*

\* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.

\* All rights reserved

/\*-----

\*/

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe\_lib\_2\_0/src/safe\_string/strlastsame\_s.c

No license file was found, but licenses were detected in source scan.

/\*-----

\* strishex\_s.c

\*

\* October 2008, Bo Berry

\*

\* Copyright (c) 2008-2010, 2013 by Cisco Systems, Inc.

\* All Rights reserved.

\*/

Found in path(s):

\* /opt/cola/permits/1502360938\_1670536039.5216763/0/safe-lib-2-0-1-tar-gz/safe\_lib\_2\_0/src/safe\_string/strishex\_s.c

## 1.26 kotlin 1.6.10

### 1.26.1 Available under license :

No license file was found, but licenses were detected in source scan.

```
{"version":3,"file":"kotlin.js","sources":["wrapper.js","js/arrayUtils.js","js/callableReferenceUtils.js","js/conversions.js","js/core.js","js/long.js","js/markerFunctions.js","js/misc.js","js/polyfills.js","js/rtti.js","runtime/arrayUtils.kt","runtime/Enum.kt","primitiveCompanionObjects.kt","common/src/generated/_Arrays.kt","common/src/generated/_Ranges.kt","unsigned/src/kotlin/UByte.kt","unsigned/src/kotlin/UInt.kt","unsigned/src/kotlin/UShort.kt","builtin-sources/Ranges.kt","src/kotlin/collections/Collections.kt","src/kotlin/collections/Maps.kt","src/kotlin/collections/Sets.kt","src/kotlin/text/StringNumberConversions.kt","src/kotlin/time/Duration.kt","unsigned/src/kotlin/UnsignedUtils.kt","src/kotlin/collections/Iterables.kt","src/kotlin/collections/Sequences.kt","src/kotlin/util/Preconditions.kt","js/src
```

/generated/\_ArraysJs.kt", "src/kotlin/comparisons/Comparisons.kt", "src/kotlin/util/Standard.kt", "js/src/generated/\_ComparisonsJs.kt", "unsigned/src/kotlin/ULong.kt", "common/src/generated/\_Collections.kt", "js/src/kotlin/collections.kt", "src/kotlin/collections/Iterators.kt", "common/src/generated/\_Comparisons.kt", "common/src/generated/\_Maps.kt", "common/src/generated/\_OneToManyTitlecaseMappings.kt", "js/src/kotlin/text/char.kt", "js/src/kotlin/text/string.kt", "src/kotlin/text/Char.kt", "src/kotlin/CharCode.kt", "common/src/generated/\_Sequences.kt", "common/src/generated/\_Sets.kt", "common/src/generated/\_Strings.kt", "src/kotlin/text/Strings.kt", "unsigned/src/kotlin/UByteArray.kt", "unsigned/src/kotlin/UIntArray.kt", "unsigned/src/kotlin/ULongArray.kt", "unsigned/src/kotlin/UShortArray.kt", "common/src/generated/\_UArrays.kt", "common/src/generated/\_UCollections.kt", "common/src/generated/\_UComparisons.kt", "common/src/generated/\_URanges.kt", "common/src/generated/\_USequences.kt", "common/src/kotlin/ExceptionsH.kt", "common/src/kotlin/JsAnnotationsH.kt", "common/src/kotlin/ioH.kt", "builtin-sources/Collections.kt", "builtin-sources/Iterators.kt", "builtin-sources/ProgressionIterators.kt", "builtin-sources/Progressions.kt", "builtin-sources/Range.kt", "builtin-sources/Unit.kt", "builtin-sources/annotation/Annotations.kt", "builtin-sources/internal/InternalAnnotations.kt", "builtin-sources/internal/progressionUtil.kt", "src/kotlin/builtins.kt", "src/kotlin/jsTypeOf.kt", "src/kotlin/kotlin.kt", "src/kotlin/CharCode\_js-v1.kt", "src/kotlin/coroutines/CoroutineImpl.kt", "src/kotlin/util/Result.kt", "src/kotlin/coroutines/Continuation.kt", "src/kotlin/coroutines/intrinsics/IntrinsicsJs.kt", "src/kotlin/currentBeMisc.kt", "src/kotlin/exceptions.kt", "src/kotlin/jsOperators.kt", "src/kotlin/math\_js-v1.kt", "src/kotlin/numbers\_js-v1.kt", "src/kotlin/reflection\_js-v1.kt", "src/kotlin/text/numberConversions\_js-v1.kt", "js/src/generated/\_CharCategories.kt", "js/src/generated/\_CollectionsJs.kt", "js/src/generated/\_DigitChars.kt", "js/src/generated/\_LetterChars.kt", "js/src/generated/\_OtherLowercaseChars.kt", "js/src/generated/\_OtherUppercaseChars.kt", "js/src/generated/\_StringsJs.kt", "js/src/generated/\_TitlecaseMappings.kt", "js/src/generated/\_UArraysJs.kt", "js/src/generated/\_WhitespaceChars.kt", "js/src/kotlin/Comparator.kt", "js/src/kotlin/annotations.kt", "js/src/kotlin/annotationsJVM.kt", "js/src/kotlin/collections/AbstractMutableCollection.kt", "js/src/kotlin/collections/AbstractMutableList.kt", "js/src/kotlin/collections/AbstractMutableMap.kt", "js/src/kotlin/collections/AbstractMutableSet.kt", "js/src/kotlin/collections/ArrayList.kt", "js/src/kotlin/collections/ArraySorting.kt", "js/src/kotlin/collections/ArraysJs.kt", "js/src/kotlin/collections/EqualityComparator.kt", "js/src/kotlin/collections/HashMap.kt", "js/src/kotlin/collections/HashSet.kt", "js/src/kotlin/collections/InternalHashMap.kt", "js/src/kotlin/collections/InternalMap.kt", "js/src/kotlin/collections/InternalStringMap.kt", "js/src/kotlin/collections/LinkedHashMap.kt", "js/src/kotlin/collections/LinkedHashSet.kt", "js/src/kotlin/concurrent.kt", "js/src/kotlin/console.kt", "js/src/kotlin/coroutines/SafeContinuationJs.kt", "js/src/kotlin/coroutines/cancellation/CancellationException.kt", "js/src/kotlin/coroutines/js/internal/EmptyContinuation.kt", "js/src/kotlin/date.kt", "js/src/kotlin/dom/Builders.kt", "js/src/kotlin/dom/Classes.kt", "js/src/kotlin/dom/Dom.kt", "js/src/kotlin/dom/EventListener.kt", "js/src/kotlin/dom/ItemArrayLike.kt", "js/src/kotlin/dom/Mutations.kt", "js/src/kotlin/dynamic.kt", "js/src/kotlin/exceptionUtils.kt", "js/src/kotlin/grouping.kt", "src/kotlin/collections/Grouping.kt", "js/src/kotlin/jsOn.kt", "js/src/kotlin/math.kt", "js/src/kotlin/numbers.kt", "js/src/kotlin/promise.kt", "js/src/kotlin/random/PlatformRandom.kt", "js/src/kotlin/reflect/AssociatedObjects.kt", "js/src/kotlin/reflect/JsClass.kt", "js/src/kotlin/reflect/KClassImpl.kt", "js/src/kotlin/reflect/KClassesImpl.kt", "js/src/kotlin/reflect/KTypeHelpers.kt", "js/src/kotlin/reflect/KTypeImpl.kt", "js/src/kotlin/reflect/KTypeParameterImpl.kt", "js/src/kotlin/reflect/primitives.kt", "js/src/kotlin/reflect/reflection.kt", "js/src/kotlin/regexp.kt", "js/src/kotlin/sequence.kt", "js/src/kotlin/text/CharCategoryJS.kt", "js/src/kotlin/text/CharacterCodingExceptionJs.kt", "js/src/kotlin/text/StringBuilderJs.kt", "js/src/kotlin/text/numberConversions.kt", "js/src/kotlin/text/regext.kt", "src/kotlin/text/StringBuilder.kt", "js/src/kotlin/text/stringsCode.kt", "js/src/kotlin/text/utf8Encoding.kt", "js/src/kotlin/throwableExtensions.kt", "js/src/kotlin/time/DurationJs.kt", "js/src/kotlin/time/DurationUnit.kt", "js/src/kotlin/time/MonoTimeSource.kt", "js/src/kotlinx/dom/Builders.kt", "js/src/kotlinx/dom/Classes.kt", "src/kotlin/text/regex/RegexExtensions.kt", "js/src/kotlinx/dom/Dom.kt", "js/src/kotlinx/dom/Mutations.kt", "js/src/org.w3c/deprecated.kt", "js/src/org.w3c/org.khronos.webgl.kt", "js/src/org.w3c/org.w3c.dom.clipboard.kt", "js/src/org.w3c/org.w3c.dom.css.kt", "js/src/org.w3c/org.w3c.dom.encryptedmedia.kt", "js/src/org.w3c/org.w3c.dom.events.kt", "js/src/org.w3c/org.w3c.dom.kt", "js/src/org.w3c/org.w3c.fetch.kt", "js/src/org.w3c/org.w3c.dom.mediacapture.kt", "js/src/org.w3c/org.w3c.dom.mediasource.kt", "js/src/org.w3c/org.w3c.dom.pointerevents.kt", "js/src/org.w3c/org.w3c.dom.svg.kt", "js/s

rc/org.w3c/org.w3c.files.kt", "js/src/org.w3c/org.w3c.notifications.kt", "js/src/org.w3c/org.w3c.workers.kt", "js/src/org.w3c/org.w3c.xhr.kt", "src/kotlin/annotations/Experimental.kt", "src/kotlin/annotations/ExperimentalStdlibApi.kt", "src/kotlin/annotations/Inference.kt", "src/kotlin/annotations/Multiplatform.kt", "src/kotlin/annotations/OptIn.kt", "src/kotlin/collections/AbstractCollection.kt", "src/kotlin/collections/AbstractIterator.kt", "src/kotlin/collections/AbstractList.kt", "src/kotlin/collections/AbstractMap.kt", "src/kotlin/collections/AbstractSet.kt", "src/kotlin/collections/ArrayDeque.kt", "src/kotlin/collections/Arrays.kt", "src/kotlin/collections/BrittleContainsOptimization.kt", "src/kotlin/collections/IndexedValue.kt", "src/kotlin/collections/MapAccessors.kt", "src/kotlin/collections/MapWithDefault.kt", "src/kotlin/collections/MutableCollections.kt", "src/kotlin/collections/ReversedViews.kt", "src/kotlin/collections/SequenceBuilder.kt", "src/kotlin/collections/SlidingWindow.kt", "src/kotlin/collections/UArraySorting.kt", "src/kotlin/comparisons/compareTo.kt", "src/kotlin/contracts/ContractBuilder.kt", "src/kotlin/coroutines/ContinuationInterceptor.kt", "src/kotlin/coroutines/CoroutineContext.kt", "src/kotlin/coroutines/CoroutineContextImpl.kt", "src/kotlin/coroutines/intrinsics/Intrinsics.kt", "src/kotlin/experimental/bitwiseOperations.kt", "src/kotlin/experimental/inferenceMarker.kt", "src/kotlin/internal/Annotations.kt", "src/kotlin/properties/Delegates.kt", "src/kotlin/properties/Interfaces.kt", "src/kotlin/properties/ObservableProperty.kt", "src/kotlin/properties/PropertyReferenceDelegates.kt", "src/kotlin/random/Random.kt", "src/kotlin/random/URandom.kt", "src/kotlin/random/XorWowRandom.kt", "src/kotlin/ranges/Ranges.kt", "src/kotlin/reflect/KClasses.kt", "src/kotlin/reflect/KTypeProjection.kt", "src/kotlin/reflect/KVariance.kt", "src/kotlin/reflect/typeOf.kt", "src/kotlin/text/Appendable.kt", "src/kotlin/text/Indent.kt", "src/kotlin/text/Typography.kt", "src/kotlin/text/regex/MatchResult.kt", "src/kotlin/time/DurationUnit.kt", "src/kotlin/time/ExperimentalTime.kt", "src/kotlin/time/TimeSource.kt", "src/kotlin/time/TimeSources.kt", "src/kotlin/time/measureTime.kt", "src/kotlin/util/DeepRecursive.kt", "src/kotlin/util/FloorDivMod.kt", "src/kotlin/util/HashCode.kt", "src/kotlin/util/KotlinVersion.kt", "src/kotlin/util/Lateinit.kt", "src/kotlin/util/Lazy.kt", "src/kotlin/util/Numbers.kt", "src/kotlin/util/Suspend.kt", "src/kotlin/util/Tuples.kt", "unsigned/src/kotlin/UIntRange.kt", "unsigned/src/kotlin/UIterators.kt", "unsigned/src/kotlin/ULongRange.kt", "unsigned/src/kotlin/UMath.kt", "unsigned/src/kotlin/UNumbers.kt", "unsigned/src/kotlin/UProgressionUtil.kt", "unsigned/src/kotlin/UStrings.kt", "unsigned/src/kotlin/annotations/Unsigned.kt", "common/src/kotlin/MathH.kt"], "sourcesContent": ["(function (root, factory) {\n if (typeof define === 'function' && define.amd) {\n define('kotlin', ['exports'], factory);\n }\n else if (typeof exports === 'object') {\n factory(module.exports);\n }\n else {\n root.kotlin = {};\n factory(root.kotlin);\n }\n})(this, function (Kotlin) {\n var \_ = Kotlin;\n insertContent();\n});\n", "/\*\n \* Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n \* Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n \*\nKotlin.isBooleanArray = function (a) {\n return (Array.isArray(a) || a instanceof Int8Array) && a.\$type\$ === \"BooleanArray\";\n};\nKotlin.isByteArray = function (a) {\n return a instanceof Int8Array && a.\$type\$ !== \"BooleanArray\";\n};\nKotlin.isShortArray = function (a) {\n return a instanceof Int16Array;\n};\nKotlin.isCharArray = function (a) {\n return a instanceof Uint16Array && a.\$type\$ === \"CharArray\";\n};\nKotlin.isIntArray = function (a) {\n return a instanceof Int32Array;\n};\nKotlin.isFloatArray = function (a) {\n return a instanceof Float32Array;\n};\nKotlin.isDoubleArray = function (a) {\n return a instanceof Float64Array;\n};\nKotlin.isLongArray = function (a) {\n return Array.isArray(a) && a.\$type\$ === \"LongArray\";\n};\nKotlin.isArray = function (a) {\n return Array.isArray(a) && !a.\$type\$;\n};\nKotlin.isArrayish = function (a) {\n return Array.isArray(a) || ArrayBuffer.isView(a);\n};\nKotlin.arrayToString = function (a) {\n if (a === null) return \"null\";\n var toString = Kotlin.isCharArray(a) ? String.fromCharCode : Kotlin.toString;\n return \"[\" + Array.prototype.map.call(a, function(e) { return toString(e); }).join(\", \") + \"]\";\n};\nKotlin.arrayDeepToString = function (arr) {\n return Kotlin.kotlin.collections.contentDeepToStringImpl(arr);\n};\nKotlin.arrayEquals = function (a, b) {\n if (a === b) {\n return true;\n }\n if (a === null || b === null || !Kotlin.isArrayish(b) || a.length !== b.length) {\n return false;\n }\n for (var i = 0, n = a.length; i < n; i++) {\n if (!Kotlin.equals(a[i], b[i])) {\n return false;\n }\n }\n return true;\n};\nKotlin.arrayDeepEquals = function (a, b) {\n return Kotlin.kotlin.collections.contentDeepEqualsImpl(a, b);\n};\nKotlin.arrayHashCode = function (arr) {\n if (arr === null) return 0;\n var result = 1;\n for (var i = 0, n = arr.length; i < n; i++) {\n

```

result = ((31 * result | 0) + Kotlin.hashCode(arr[i])) | 0;\n    }\n    return result;\n};\n\nKotlin.arrayDeepHashCode =
function (arr) {\n    return
Kotlin.kotlin.collections.contentDeepHashCodeImpl(arr);\n};\n\nKotlin.primitiveArraySort = function (array) {\n
array.sort(Kotlin.doubleCompareTo);\n};\n\n"/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming
Language contributors. \n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\nKotlin.getCallableRef = function(name, f) {\n    f.callableName = name;\n    return
f;\n};\n\nKotlin.getPropertyCallableRef = function(name, paramCount, getter, setter) {\n    getter.get = getter;\n
getter.set = setter;\n    getter.callableName = name;\n    return getPropertyRefClass(getter, setter,
propertyRefClassMetadataCache[paramCount]);\n};\n\nfunction getPropertyRefClass(obj, setter, cache) {\n
obj.$metadata$ = getPropertyRefMetadata(typeof setter === "function" ? cache.mutable : cache.immutable);\n
obj.constructor = obj;\n    return obj;\n};\n\nvar propertyRefClassMetadataCache = [\n    {\n        mutable: { value:
null, implementedInterface: function () {\n            return Kotlin.kotlin.reflect.KMutableProperty0 }\n        },\n
immutable: { value: null, implementedInterface: function () {\n            return Kotlin.kotlin.reflect.KProperty0 }\n
        }\n    },\n    {\n        mutable: { value: null, implementedInterface: function () {\n            return
Kotlin.kotlin.reflect.KMutableProperty1 }\n        },\n        immutable: { value: null, implementedInterface: function
() {\n            return Kotlin.kotlin.reflect.KProperty1 }\n        }\n    };\n\nfunction getPropertyRefMetadata(cache)
{\n    if (cache.value === null) {\n        cache.value = {\n            interfaces: [cache.implementedInterface()],\n
baseClass: null,\n            functions: {},\n            properties: {},\n            types: {},\n            staticMembers: {}
}\n    }\n    return cache.value;\n};\n\n"/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming
Language contributors. \n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\nKotlin.toShort = function (a) {\n    return (a & 0xFFFF) << 16 >>
16;\n};\n\nKotlin.toByte = function (a) {\n    return (a & 0xFF) << 24 >> 24;\n};\n\nKotlin.toChar = function (a) {\n
return a & 0xFFFF;\n};\n\nKotlin.numberToLong = function (a) {\n    return a instanceof Kotlin.Long ? a :
Kotlin.Long.fromNumber(a);\n};\n\nKotlin.numberToInt = function (a) {\n    return a instanceof Kotlin.Long ?
a.toInt() : Kotlin.doubleToInt(a);\n};\n\nKotlin.numberToShort = function (a) {\n    return
Kotlin.toShort(Kotlin.numberToInt(a));\n};\n\nKotlin.numberToByte = function (a) {\n    return
Kotlin.toByte(Kotlin.numberToInt(a));\n};\n\nKotlin.numberToDouble = function (a) {\n    return
+a;\n};\n\nKotlin.numberToChar = function (a) {\n    return
Kotlin.toChar(Kotlin.numberToInt(a));\n};\n\nKotlin.doubleToInt = function(a) {\n    if (a > 2147483647) return
2147483647;\n    if (a < -2147483648) return -2147483648;\n    return a | 0;\n};\n\nKotlin.toBoxedChar = function
(a) {\n    if (a == null) return a;\n    if (a instanceof Kotlin.BoxedChar) return a;\n    return new
Kotlin.BoxedChar(a);\n};\n\nKotlin.unboxChar = function(a) {\n    if (a == null) return a;\n    return
Kotlin.toChar(a);\n};\n\n"/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
contributors. \n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\nKotlin.equals = function (obj1, obj2) {\n    if (obj1 == null) {\n        return obj2 ==
null;\n    }\n\n    if (obj2 == null) {\n        return false;\n    }\n\n    if (obj1 !== obj2) {\n        return obj2 !== obj2;\n
    }\n\n    if (typeof obj1 === "object" && typeof obj1.equals === "function") {\n        return obj1.equals(obj2);\n
    }\n\n    if (typeof obj1 === "number" && typeof obj2 === "number") {\n        return obj1 === obj2 && (obj1 !==
0 || 1 / obj1 === 1 / obj2);\n    }\n\n    return obj1 === obj2;\n};\n\nKotlin.hashCode = function (obj) {\n    if (obj ==
null) {\n        return 0;\n    }\n\n    var objType = typeof obj;\n    if ("object" === objType) {\n        return "function"
=== typeof obj.hashCode ? obj.hashCode() : getObjectHashCode(obj);\n    }\n\n    if ("function" === objType) {\n
return getObjectHashCode(obj);\n    }\n\n    if ("number" === objType) {\n        return
Kotlin.numberHashCode(obj);\n    }\n\n    if ("boolean" === objType) {\n        return Number(obj)\n    }\n\n    var str
= String(obj);\n    return getStringHashCode(str);\n};\n\n\nKotlin.toString = function (o) {\n    if (o == null) {\n
return "null";\n    }\n\n    else if (Kotlin.isArrayish(o)) {\n        return "["+...+";\n    }\n\n    else {\n        return
o.toString();\n    }\n};\n\n\n/** @const */\nvar POW_2_32 = 4294967296;\n\n// TODO: consider switching to Symbol
type once we are on ES6.\n\n/** @const */\nvar OBJECT_HASH_CODE_PROPERTY_NAME =
"\"kotlinHashCodeValue$\";\n\nfunction getObjectHashCode(obj) {\n    if

```

```

(! (OBJECT_HASH_CODE_PROPERTY_NAME in obj)) {
    var hash = (Math.random() * POW_2_32) | 0; //
    Make 32-bit signed integer.
    Object.defineProperty(obj, OBJECT_HASH_CODE_PROPERTY_NAME, {
        value: hash, enumerable: false });
}
return
obj[OBJECT_HASH_CODE_PROPERTY_NAME];
}
function getStringHashCode(str) {
    var hash = 0;
    for (var i = 0; i < str.length; i++) {
        var code = str.charCodeAt(i);
        hash = (hash * 31 + code) | 0; // Keep
        it 32-bit.
    }
    return hash;
}
Kotlin.identityHashCode = getObjectHashCode;
"/**
 * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.
 * Use of this source code is governed by the Apache 2.0 license that can be
    found in the license/LICENSE.txt file.
 */
Copyright 2009 The Closure Library Authors. All Rights Reserved.
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0
Unless required by applicable law or agreed to in writing,
software distributed under the License is distributed on an "AS-IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
*/
Constructs a 64-bit two's-complement integer, given its low and high 32-bit
values as signed integers. See the from* functions below for more
convenient ways of constructing Longs.
The internal representation of a long is the two given signed, 32-bit
values. We use 32-bit pieces because these are the size of integers on which
Javascript performs bit-operations. For operations like addition and
multiplication, we split each number into 16-bit pieces, which can easily be
multiplied within Javascript's floating-point representation without overflow
or change in sign.
In the algorithms below, we frequently reduce the negative case to the
positive case by negating the input(s) and then post-processing the result.
Note that we must ALWAYS check specially whether those values are
MIN_VALUE (-2^63) because -MIN_VALUE == MIN_VALUE (since 2^63 cannot be
represented as a positive number, it overflows back into a negative). Not
handling this case would often result in infinite recursion.
@param {number} low The low (signed) 32 bits of the long.
@param {number} high The high (signed) 32 bits of the long.
@constructor
@final
Kotlin.Long = function(low, high) {
    /**
     * @type {number}
     * @private
     */
    this.low_ = low | 0; // force into 32 signed bits.
    /**
     * @type {number}
     * @private
     */
    this.high_ = high | 0; // force into 32 signed bits.
};
Kotlin.Long.$metadata$ = {
    kind: "class",
    simpleName: "Long",
    interfaces: []
};
NOTE: Common constant values ZERO, ONE, NEG_ONE, etc. are defined
below the from* methods on which they depend.
*/
A cache of the Long representations of small integer values.
@type {!Object}
@private
Kotlin.Long.IntCache_ = {};
Returns a Long representing the given (32-bit) integer value.
@param {number} value The 32-bit integer in question.
@return {!Kotlin.Long} The corresponding Long value.
Kotlin.Long.fromInt = function(value) {
    if (-128 <= value && value < 128) {
        var cachedObj = Kotlin.Long.IntCache_[value];
        if (cachedObj) {
            return cachedObj;
        }
        var obj = new Kotlin.Long(value | 0, value < 0 ? -1 : 0);
        if (-128 <= value && value < 128) {
            Kotlin.Long.IntCache_[value] = obj;
        }
        return obj;
    }
    Converts this number value to `Long`. The fractional part, if any, is rounded
    down towards zero. Returns zero if this `Double` value is `NaN`,
    `Long.MIN_VALUE` if it's less than `Long.MIN_VALUE`, `Long.MAX_VALUE`
    if it's bigger than `Long.MAX_VALUE`.
    @param {number} value The number in question.
    @return {!Kotlin.Long} The corresponding Long value.
Kotlin.Long.fromNumber = function(value) {
    if (isNaN(value)) {
        return Kotlin.Long.ZERO;
    } else if (value <= -
        Kotlin.Long.TWO_PWR_63_DBL_) {
        return Kotlin.Long.MIN_VALUE;
    } else if (value + 1 >=
        Kotlin.Long.TWO_PWR_63_DBL_) {
        return Kotlin.Long.MAX_VALUE;
    } else if (value < 0) {
        return Kotlin.Long.fromNumber(-value).negate();
    } else {
        return new Kotlin.Long(
            (value % Kotlin.Long.TWO_PWR_32_DBL_) | 0,
            (value / Kotlin.Long.TWO_PWR_32_DBL_) | 0);
    }
};
Returns a Long representing the 64-bit integer that comes by concatenating
the given high and low bits. Each is assumed to use 32 bits.
@param {number} lowBits The low 32-bits.
@param {number} highBits The high 32-bits.
@return {!Kotlin.Long} The corresponding Long value.
Kotlin.Long.fromBits = function(lowBits, highBits) {
    return new Kotlin.Long(lowBits, highBits);
};
Returns a Long representation of the given string, written using the given
radix.
@param {string} str The textual

```

```

representation of the Long.\n * @param {number=} opt_radix The radix in which the text is written.\n * @return
{!Kotlin.Long} The corresponding Long value.\n */\nKotlin.Long.fromString = function(str, opt_radix) {\n if
(str.length == 0) {\n throw Error('number format error: empty string');\n }\n\n var radix = opt_radix || 10;\n if
(radix < 2 || 36 < radix) {\n throw Error('radix out of range: ' + radix);\n }\n\n if (str.charAt(0) == '-') {\n return
Kotlin.Long.fromString(str.substring(1), radix).negate();\n } else if (str.indexOf('-') >= 0) {\n throw Error('number
format error: interior \"-\" character: ' + str);\n }\n\n // Do several (8) digits each time through the loop, so as to\n //
minimize the calls to the very expensive emulated div.\n var radixToPower =
Kotlin.Long.fromNumber(Math.pow(radix, 8));\n var result = Kotlin.Long.ZERO;\n for (var i = 0; i < str.length;
i += 8) {\n var size = Math.min(8, str.length - i);\n var value = parseInt(str.substring(i, i + size), radix);\n if
(size < 8) {\n var power = Kotlin.Long.fromNumber(Math.pow(radix, size));\n result =
result.multiply(power).add(Kotlin.Long.fromNumber(value));\n } else {\n result =
result.multiply(radixToPower);\n result = result.add(Kotlin.Long.fromNumber(value));\n }\n }\n return
result;\n};\n\n// NOTE: the compiler should inline these constant values below and then remove\n// these
variables, so there should be no runtime penalty for these.\n\n/**\n * Number used repeated below in calculations.
This must appear before the\n * first call to any from* function below.\n * @type {number}\n * @private\n
*/\nKotlin.Long.TWO_PWR_16_DBL_ = 1 << 16;\n\n/**\n * @type {number}\n * @private\n
*/\nKotlin.Long.TWO_PWR_24_DBL_ = 1 << 24;\n\n/**\n * @type {number}\n * @private\n
*/\nKotlin.Long.TWO_PWR_32_DBL_ =\n Kotlin.Long.TWO_PWR_16_DBL_ *
Kotlin.Long.TWO_PWR_16_DBL_;\n\n/**\n * @type {number}\n * @private\n
*/\nKotlin.Long.TWO_PWR_31_DBL_ =\n Kotlin.Long.TWO_PWR_32_DBL_ / 2;\n\n/**\n * @type
{number}\n * @private\n */\nKotlin.Long.TWO_PWR_48_DBL_ =\n Kotlin.Long.TWO_PWR_32_DBL_ *
Kotlin.Long.TWO_PWR_16_DBL_;\n\n/**\n * @type {number}\n * @private\n
*/\nKotlin.Long.TWO_PWR_64_DBL_ =\n Kotlin.Long.TWO_PWR_32_DBL_ *
Kotlin.Long.TWO_PWR_32_DBL_;\n\n/**\n * @type {number}\n * @private\n
*/\nKotlin.Long.TWO_PWR_63_DBL_ =\n Kotlin.Long.TWO_PWR_64_DBL_ / 2;\n\n/**\n * @type
{!Kotlin.Long}\n */\nKotlin.Long.ZERO = Kotlin.Long.fromInt(0);\n\n/**\n * @type {!Kotlin.Long}\n
*/\nKotlin.Long.ONE = Kotlin.Long.fromInt(1);\n\n/**\n * @type {!Kotlin.Long}\n */\nKotlin.Long.NEG_ONE =
Kotlin.Long.fromInt(-1);\n\n/**\n * @type {!Kotlin.Long}\n */\nKotlin.Long.MAX_VALUE =\n
Kotlin.Long.fromBits(0xFFFFFFFF | 0, 0x7FFFFFFF | 0);\n\n/**\n * @type {!Kotlin.Long}\n
*/\nKotlin.Long.MIN_VALUE = Kotlin.Long.fromBits(0, 0x80000000 | 0);\n\n/**\n * @type {!Kotlin.Long}\n *
@private\n */\nKotlin.Long.TWO_PWR_24_ = Kotlin.Long.fromInt(1 << 24);\n\n/**\n * @return {number} The
value, assuming it is a 32-bit integer. */\nKotlin.Long.prototype.toInt = function() {\n return this.low_;\n};\n\n/**\n
* @return {number} The closest floating-point representation to this value. */\nKotlin.Long.prototype.toNumber =
function() {\n return this.high_ * Kotlin.Long.TWO_PWR_32_DBL_ +\n
this.getLowBitsUnsigned();\n};\n\n/**\n * @return {number} The 32-bit hashCode of this value.
*/\nKotlin.Long.prototype.hashCode = function() {\n return this.high_ ^ this.low_;\n};\n\n/**\n * @param
{number=} opt_radix The radix in which the text should be written.\n * @return {string} The textual representation
of this value.\n * @override\n */\nKotlin.Long.prototype.toString = function(opt_radix) {\n var radix = opt_radix ||
10;\n if (radix < 2 || 36 < radix) {\n throw Error('radix out of range: ' + radix);\n }\n\n if (this.isZero()) {\n
return '0';\n }\n\n if (this.isNegative()) {\n if (this.equalsLong(Kotlin.Long.MIN_VALUE)) {\n // We need to
change the Long value before it can be negated, so we remove\n // the bottom-most digit in this base and then
recurse to do the rest.\n var radixLong = Kotlin.Long.fromNumber(radix);\n var div = this.div(radixLong);\n
var rem = div.multiply(radixLong).subtract(this);\n return div.toString(radix) + rem.toInt().toString(radix);\n }
else {\n return '-' + this.negate().toString(radix);\n }\n }\n\n // Do several (6) digits each time through the loop,
so as to\n // minimize the calls to the very expensive emulated div.\n var radixToPower =
Kotlin.Long.fromNumber(Math.pow(radix, 6));\n var rem = this;\n var result = '';\n while (true) {\n var
remDiv = rem.div(radixToPower);\n var intVal = rem.subtract(remDiv.multiply(radixToPower)).toInt();\n var
digits = intVal.toString(radix);\n rem = remDiv;\n if (rem.isZero()) {\n return digits + result;\n } else {\n

```

```

while (digits.length < 6) {\n    digits = '0' + digits;\n    } \n    result = " + digits + result;\n    }\n }\n};\n\n\n/**
@return {number} The high 32-bits as a signed value. */\nKotlin.Long.prototype.getHighBits = function() {\n
return this.high_;\n};\n\n\n/**
@return {number} The low 32-bits as a signed value.
*/\nKotlin.Long.prototype.getLowBits = function() {\n    return this.low_;\n};\n\n\n/**
@return {number} The low
32-bits as an unsigned value. */\nKotlin.Long.prototype.getLowBitsUnsigned = function() {\n    return (this.low_ >=
0) ?\n    this.low_ : Kotlin.Long.TWO_PWR_32_DBL_ + this.low_;\n};\n\n\n\n/**
*n * @return {number} Returns
the number of bits needed to represent the absolute\n *    value of this Long.\n
*/\nKotlin.Long.prototype.getNumBitsAbs = function() {\n    if (this.isNegative()) {\n        if
(this.equalsLong(Kotlin.Long.MIN_VALUE)) {\n            return 64;\n        } else {\n            return
this.negate().getNumBitsAbs();\n        }\n    } else {\n        var val = this.high_ != 0 ? this.high_ : this.low_;\n        for (var bit
= 31; bit > 0; bit--) {\n            if ((val & (1 << bit)) != 0) {\n                break;\n            }\n        }\n        return this.high_ != 0 ? bit + 33
: bit + 1;\n    }\n};\n\n\n\n/**
@return {boolean} Whether this value is zero.
*/\nKotlin.Long.prototype.isZero =
function() {\n    return this.high_ == 0 && this.low_ == 0;\n};\n\n\n\n/**
@return {boolean} Whether this value is
negative.
*/\nKotlin.Long.prototype.isNegative = function() {\n    return this.high_ < 0;\n};\n\n\n\n/**
@return
{boolean} Whether this value is odd.
*/\nKotlin.Long.prototype.isOdd = function() {\n    return (this.low_ & 1) ==
1;\n};\n\n\n\n\n/**
*n * @param {Kotlin.Long} other Long to compare against.\n * @return {boolean} Whether this
Long equals the other.\n
*/\nKotlin.Long.prototype.equalsLong = function(other) {\n    return (this.high_ ==
other.high_) && (this.low_ == other.low_);\n};\n\n\n\n\n/**
*n * @param {Kotlin.Long} other Long to compare
against.\n * @return {boolean} Whether this Long does not equal the other.\n
*/\nKotlin.Long.prototype.notEqualsLong = function(other) {\n    return (this.high_ != other.high_) || (this.low_ !=
other.low_);\n};\n\n\n\n\n/**
*n * @param {Kotlin.Long} other Long to compare against.\n * @return {boolean}
Whether this Long is less than the other.\n
*/\nKotlin.Long.prototype.lessThan = function(other) {\n    return
this.compare(other) < 0;\n};\n\n\n\n\n/**
*n * @param {Kotlin.Long} other Long to compare against.\n * @return
{boolean} Whether this Long is less than or equal to the other.\n
*/\nKotlin.Long.prototype.lessThanOrEqual =
function(other) {\n    return this.compare(other) <= 0;\n};\n\n\n\n\n/**
*n * @param {Kotlin.Long} other Long to
compare against.\n * @return {boolean} Whether this Long is greater than the other.\n
*/\nKotlin.Long.prototype.greaterThan = function(other) {\n    return this.compare(other) > 0;\n};\n\n\n\n\n/**
*n * @param {Kotlin.Long} other Long to compare against.\n * @return {boolean} Whether this Long is greater than or
equal to the other.\n
*/\nKotlin.Long.prototype.greaterThanOrEqual = function(other) {\n    return
this.compare(other) >= 0;\n};\n\n\n\n\n\n/**
*n * @param {Kotlin.Long}
other Long to compare against.\n * @return {number} 0 if they are the same, 1 if the this is greater, and -1\n *    if
the given one is greater.\n
*/\nKotlin.Long.prototype.compare = function(other) {\n    if (this.equalsLong(other)) {\n
return 0;\n    }\n    var thisNeg = this.isNegative();\n    var otherNeg = other.isNegative();\n    if (thisNeg &&
!otherNeg) {\n        return -1;\n    }\n    if (!thisNeg && otherNeg) {\n        return 1;\n    }\n    // at this point, the signs are the
same, so subtraction will not overflow\n    if (this.subtract(other).isNegative()) {\n        return -1;\n    } else {\n        return
1;\n    }\n};\n\n\n\n\n/**
@return {!Kotlin.Long} The negation of this value.
*/\nKotlin.Long.prototype.negate =
function() {\n    if (this.equalsLong(Kotlin.Long.MIN_VALUE)) {\n        return Kotlin.Long.MIN_VALUE;\n    } else
{\n        return this.not().add(Kotlin.Long.ONE);\n    }\n};\n\n\n\n\n/**
*n * Returns the sum of this and the given Long.\n *
@param {Kotlin.Long} other Long to add to this one.\n * @return {!Kotlin.Long} The sum of this and the given
Long.\n
*/\nKotlin.Long.prototype.add = function(other) {\n    // Divide each number into 4 chunks of 16 bits, and
then sum the chunks.\n    var a48 = this.high_ >>> 16;\n    var a32 = this.high_ & 0xFFFF;\n    var a16 = this.low_
>>> 16;\n    var a00 = this.low_ & 0xFFFF;\n    var b48 = other.high_ >>> 16;\n    var b32 = other.high_ & 0xFFFF;\n
var b16 = other.low_ >>> 16;\n    var b00 = other.low_ & 0xFFFF;\n    var c48 = 0, c32 = 0, c16 = 0, c00 = 0;\n    c00
+= a00 + b00;\n    c16 += c00 >>> 16;\n    c00 &= 0xFFFF;\n    c16 += a16 + b16;\n    c32 += c16 >>> 16;\n    c16 &=
0xFFFF;\n    c32 += a32 + b32;\n    c48 += c32 >>> 16;\n    c32 &= 0xFFFF;\n    c48 += a48 + b48;\n    c48 &=
0xFFFF;\n    return Kotlin.Long.fromBits((c16 << 16) | c00, (c48 << 16) | c32);\n};\n\n\n\n\n/**
*n * Returns the
difference of this and the given Long.\n * @param {Kotlin.Long} other Long to subtract from this.\n * @return
{!Kotlin.Long} The difference of this and the given Long.\n
*/\nKotlin.Long.prototype.subtract = function(other)

```

```

{\n return this.add(other.negate());\n};\n\n\n/**\n * Returns the product of this and the given long.\n * @param\n * {Kotlin.Long} other Long to multiply with this.\n * @return {!Kotlin.Long} The product of this and the other.\n */\nKotlin.Long.prototype.multiply = function(other) {\n if (this.isZero()) {\n return Kotlin.Long.ZERO;\n } else\n if (other.isZero()) {\n return Kotlin.Long.ZERO;\n }\n\n if (this.equalsLong(Kotlin.Long.MIN_VALUE)) {\n\n return other.isOdd() ? Kotlin.Long.MIN_VALUE : Kotlin.Long.ZERO;\n } else if\n (other.equalsLong(Kotlin.Long.MIN_VALUE)) {\n return this.isOdd() ? Kotlin.Long.MIN_VALUE :\n Kotlin.Long.ZERO;\n }\n\n if (this.isNegative()) {\n if (other.isNegative()) {\n return\n this.negate().multiply(other.negate());\n } else {\n return this.negate().multiply(other).negate();\n }\n } else if\n (other.isNegative()) {\n return this.multiply(other.negate()).negate();\n }\n\n // If both longs are small, use float\n multiplication\n if (this.lessThan(Kotlin.Long.TWO_PWR_24_) &&\n other.lessThan(Kotlin.Long.TWO_PWR_24_)) {\n return Kotlin.Long.fromNumber(this.toNumber() *\n other.toNumber());\n }\n\n // Divide each long into 4 chunks of 16 bits, and then add up 4x4 products.\n // We can\n skip products that would overflow.\n\n var a48 = this.high_ >>> 16;\n var a32 = this.high_ & 0xFFFF;\n var a16 =\n this.low_ >>> 16;\n var a00 = this.low_ & 0xFFFF;\n\n var b48 = other.high_ >>> 16;\n var b32 = other.high_ &\n 0xFFFF;\n var b16 = other.low_ >>> 16;\n var b00 = other.low_ & 0xFFFF;\n\n var c48 = 0, c32 = 0, c16 = 0, c00\n = 0;\n c00 += a00 * b00;\n c16 += c00 >>> 16;\n c00 &= 0xFFFF;\n c16 += a16 * b00;\n c32 += c16 >>> 16;\n c16 &= 0xFFFF;\n c16 += a00 * b16;\n c32 += c16 >>> 16;\n c16 &= 0xFFFF;\n c32 += a32 * b00;\n c48 +=\n c32 >>> 16;\n c32 &= 0xFFFF;\n c32 += a16 * b16;\n c48 += c32 >>> 16;\n c32 &= 0xFFFF;\n c32 += a00 *\n b32;\n c48 += c32 >>> 16;\n c32 &= 0xFFFF;\n c48 += a48 * b00 + a32 * b16 + a16 * b32 + a00 * b48;\n c48\n &= 0xFFFF;\n return Kotlin.Long.fromBits((c16 << 16) | c00, (c48 << 16) | c32);\n};\n\n\n/**\n * Returns this\n Long divided by the given one.\n * @param {Kotlin.Long} other Long by which to divide.\n * @return\n * {!Kotlin.Long} This Long divided by the given one.\n */\nKotlin.Long.prototype.div = function(other) {\n if\n (other.isZero()) {\n throw Error('division by zero');\n } else if (this.isZero()) {\n return Kotlin.Long.ZERO;\n }\n\n if (this.equalsLong(Kotlin.Long.MIN_VALUE)) {\n if (other.equalsLong(Kotlin.Long.ONE) ||\n other.equalsLong(Kotlin.Long.NEG_ONE)) {\n return Kotlin.Long.MIN_VALUE; // recall that -MIN_VALUE\n == MIN_VALUE\n } else if (other.equalsLong(Kotlin.Long.MIN_VALUE)) {\n return Kotlin.Long.ONE;\n }\n } else {\n // At this point, we have |other| >= 2, so |this/other| < |MIN_VALUE|.
var halfThis =\n this.shiftRight(1);\n var approx = halfThis.div(other).shiftLeft(1);\n if\n (approx.equalsLong(Kotlin.Long.ZERO)) {\n return other.isNegative() ? Kotlin.Long.ONE :\n Kotlin.Long.NEG_ONE;\n } else {\n var rem = this.subtract(other.multiply(approx));\n var result =\n approx.add(rem.div(other));\n return result;\n }\n }\n } else if\n (other.equalsLong(Kotlin.Long.MIN_VALUE)) {\n return Kotlin.Long.ZERO;\n }\n\n if (this.isNegative()) {\n\n if (other.isNegative()) {\n return this.negate().div(other.negate());\n } else {\n return\n this.negate().div(other).negate();\n }\n } else if (other.isNegative()) {\n return\n this.div(other.negate()).negate();\n }\n\n // Repeat the following until the remainder is less than other: find a\n // floating-point that approximates remainder / other *from below*, add this\n // into the result, and subtract it from\n the remainder. It is critical that\n // the approximate value is less than or equal to the real value so that the\n // remainder never becomes negative.\n var res = Kotlin.Long.ZERO;\n var rem = this;\n while\n (rem.greaterThanOrEqual(other)) {\n // Approximate the result of division. This may be a little greater or\n // smaller than the actual value.\n var approx = Math.max(1, Math.floor(rem.toNumber() / other.toNumber()));\n\n // We will tweak the approximate result by changing it in the 48-th digit or\n // the smallest non-fractional digit,\n whichever is larger.\n var log2 = Math.ceil(Math.log(approx) / Math.LN2);\n var delta = (log2 <= 48) ? 1 :\n Math.pow(2, log2 - 48);\n // Decrease the approximation until it is smaller than the remainder. Note\n // that if\n it is too large, the product overflows and is negative.\n var approxRes = Kotlin.Long.fromNumber(approx);\n var approxRem = approxRes.multiply(other);\n while (approxRem.isNegative() || approxRem.greaterThan(rem))\n {\n approx -= delta;\n approxRes = Kotlin.Long.fromNumber(approx);\n approxRem =\n approxRes.multiply(other);\n }\n\n // We know the answer can't be zero... and actually, zero would cause\n // infinite recursion since we would make no progress.\n if (approxRes.isZero()) {\n approxRes =

```

```

Kotlin.Long.ONE;\n };\n\n res = res.add(approxRes);\n rem = rem.subtract(approxRem);\n }\n return
res;\n};\n\n/**\n * Returns this Long modulo the given one.\n * @param {Kotlin.Long} other Long by which to
mod.\n * @return {!Kotlin.Long} This Long modulo the given one.\n */\nKotlin.Long.prototype.modulo =
function(other) {\n return this.subtract(this.div(other).multiply(other));\n};\n\n/**\n * @return {!Kotlin.Long} The
bitwise-NOT of this value. *\nKotlin.Long.prototype.not = function() {\n return Kotlin.Long.fromBits(~this.low_,
~this.high_);\n};\n\n/**\n * Returns the bitwise-AND of this Long and the given one.\n * @param {Kotlin.Long}
other The Long with which to AND.\n * @return {!Kotlin.Long} The bitwise-AND of this and the other.\n
*\nKotlin.Long.prototype.and = function(other) {\n return Kotlin.Long.fromBits(this.low_ & other.low_,\n
this.high_ & other.high_);\n};\n\n/**\n * Returns the bitwise-OR of this Long and the given one.\n *
@param {Kotlin.Long} other The Long with which to OR.\n * @return {!Kotlin.Long} The bitwise-OR of this and
the other.\n *\nKotlin.Long.prototype.or = function(other) {\n return Kotlin.Long.fromBits(this.low_ |
other.low_,\n this.high_ | other.high_);\n};\n\n/**\n * Returns the bitwise-XOR of this Long
and the given one.\n * @param {Kotlin.Long} other The Long with which to XOR.\n * @return {!Kotlin.Long}
The bitwise-XOR of this and the other.\n *\nKotlin.Long.prototype.xor = function(other) {\n return
Kotlin.Long.fromBits(this.low_ ^ other.low_,\n this.high_ ^ other.high_);\n};\n\n/**\n *
Returns this Long with bits shifted to the left by the given amount.\n * @param {number} numBits The number of
bits by which to shift.\n * @return {!Kotlin.Long} This shifted to the left by the given amount.\n
*\nKotlin.Long.prototype.shiftLeft = function(numBits) {\n numBits &= 63;\n if (numBits == 0) {\n return
this;\n } else {\n var low = this.low_;\n if (numBits < 32) {\n var high = this.high_;\n return
Kotlin.Long.fromBits(\n low << numBits,\n (high << numBits) | (low >>> (32 - numBits));\n } else
{\n return Kotlin.Long.fromBits(0, low << (numBits - 32));\n }\n };\n\n/**\n * Returns this Long with
bits shifted to the right by the given amount.\n * @param {number} numBits The number of bits by which to shift.\n
* @return {!Kotlin.Long} This shifted to the right by the given amount.\n *\nKotlin.Long.prototype.shiftRight =
function(numBits) {\n numBits &= 63;\n if (numBits == 0) {\n return this;\n } else {\n var high = this.high_;\n
if (numBits < 32) {\n var low = this.low_;\n return Kotlin.Long.fromBits(\n (low >>> numBits) | (high
<<< (32 - numBits)),\n high >> numBits);\n } else {\n return Kotlin.Long.fromBits(\n high >>
(numBits - 32),\n high >= 0 ? 0 : -1);\n }\n };\n\n/**\n * Returns this Long with bits shifted to the right
by the given amount, with\n * zeros placed into the new leading bits.\n * @param {number} numBits The number
of bits by which to shift.\n * @return {!Kotlin.Long} This shifted to the right by the given amount, with\n *
zeros placed into the new leading bits.\n *\nKotlin.Long.prototype.shiftRightUnsigned = function(numBits) {\n numBits
&= 63;\n if (numBits == 0) {\n return this;\n } else {\n var high = this.high_;\n if (numBits < 32) {\n var
low = this.low_;\n return Kotlin.Long.fromBits(\n (low >>> numBits) | (high <<< (32 - numBits)),\n
high >>> numBits);\n } else if (numBits == 32) {\n return Kotlin.Long.fromBits(high, 0);\n } else {\n
return Kotlin.Long.fromBits(high >>> (numBits - 32), 0);\n }\n };\n\n// Support for
Kotlin\nKotlin.Long.prototype.equals = function (other) {\n return other instanceof Kotlin.Long &&
this.equalsLong(other);\n};\n\nKotlin.Long.prototype.compareTo_11rb$ =
Kotlin.Long.prototype.compare;\n\nKotlin.Long.prototype.inc = function() {\n return
this.add(Kotlin.Long.ONE);\n};\n\nKotlin.Long.prototype.dec = function() {\n return
this.add(Kotlin.Long.NEG_ONE);\n};\n\nKotlin.Long.prototype.valueOf = function() {\n return
this.toNumber();\n};\n\nKotlin.Long.prototype.unaryPlus = function() {\n return
this;\n};\n\nKotlin.Long.prototype.unaryMinus = Kotlin.Long.prototype.negate;\nKotlin.Long.prototype.inv =
Kotlin.Long.prototype.not;\n\nKotlin.Long.prototype.rangeTo = function (other) {\n return new
Kotlin.kotlin.ranges.LongRange(this, other);\n};\n\n/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin
Programming Language contributors. \n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.\n */\n\n/**\n * @param {string} id\n * @param {Object} declaration\n
*\nKotlin.defineModule = function (id, declaration) {\n};\n\nKotlin.defineInlineFunction = function(tag, fun) {\n
return fun;\n};\n\nKotlin.wrapFunction = function(fun) {\n var f = function() {\n f = fun();\n return
f.apply(this, arguments);\n };\n return function() {\n return f.apply(this, arguments);\n
}

```

```

};\n\nKotlin.isTypeOf = function(type) {\n    return function (object) {\n        return typeof object === type;\n    }\n};\n\nKotlin.isInstanceOf = function (class) {\n    return function (object) {\n        return Kotlin.isType(object, class);\n    }\n};\n\nKotlin.orNull = function (fn) {\n    return function (object) {\n        return object == null || fn(object);\n    }\n};\n\nKotlin.andPredicate = function (a, b) {\n    return function (object) {\n        return a(object) && b(object);\n    }\n};\n\nKotlin.kotlinModuleMetadata = function (abiVersion, moduleName, data) {\n};\n\nKotlin.suspendCall = function(value) {\n    return value;\n};\n\nKotlin.coroutineResult = function(qualifier) {\n    throwMarkerError();\n};\n\nKotlin.coroutineController = function(qualifier) {\n    throwMarkerError();\n};\n\nKotlin.coroutineReceiver = function(qualifier) {\n    throwMarkerError();\n};\n\nKotlin.setCoroutineResult = function(value, qualifier) {\n    throwMarkerError();\n};\n\nKotlin.getReifiedTypeParameterKType = function(typeParameter) {\n    throwMarkerError();\n};\n\nfunction throwMarkerError() {\n    throw new Error("\n    \"This marker function should never be called. \" +\n        \"Looks like compiler did not eliminate it properly. \" +\n        \"Please, report an issue if you caught this exception.\");\n}\n\nKotlin.getFunctionById = function(id, defaultValue) {\n    return function() {\n        return defaultValue;\n    }\n};\n\n/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\nKotlin.compareTo = function (a, b) {\n    var typeA = typeof a;\n    if (typeA === \"number\") {\n        if (typeof b === \"number\") {\n            return Kotlin.doubleCompareTo(a, b);\n        }\n        return Kotlin.primitiveCompareTo(a, b);\n    }\n    if (typeA === \"string\" || typeA === \"boolean\") {\n        return Kotlin.primitiveCompareTo(a, b);\n    }\n    return\n\na.compareTo_11rb$(b);\n};\n\nKotlin.primitiveCompareTo = function (a, b) {\n    return a < b ? -1 : a > b ? 1 : 0;\n};\n\nKotlin.doubleCompareTo = function (a, b) {\n    if (a < b) return -1;\n    if (a > b) return 1;\n\n    if (a === b) {\n        if (a !== 0) return 0;\n\n        var ia = 1 / a;\n        return ia === 1 / b ? 0 : (ia < 0 ? -1 : 1);\n    }\n\n    return a !== a ? (b !== b ? 0 : 1) : -1;\n};\n\nKotlin.charInc = function (value) {\n    return Kotlin.toChar(value+1);\n};\n\nKotlin.charDec = function (value) {\n    return Kotlin.toChar(value-1);\n};\n\nKotlin.imul = Math.imul || imul;\n\nKotlin.imulEmulated = imul;\n\nfunction imul(a, b) {\n    return ((a & 0xffff0000) * (b & 0xffff) + (a & 0xffff) * (b | 0)) | 0;\n}\n\n(function() {\n    var buf = new ArrayBuffer(8);\n    var bufFloat64 = new Float64Array(buf);\n    var bufFloat32 = new Float32Array(buf);\n    var bufInt32 = new Int32Array(buf);\n    var lowIndex = 0;\n    var highIndex = 1;\n\n    bufFloat64[0] = -1; // bff00000_00000000\n    if (bufInt32[lowIndex] !== 0) {\n        lowIndex = 1;\n        highIndex = 0;\n    }\n\n    Kotlin.doubleToBits = function(value) {\n        return Kotlin.doubleToRawBits(isNaN(value) ? NaN : value);\n    }\n};\n\nKotlin.doubleToRawBits = function(value) {\n    bufFloat64[0] = value;\n    return Kotlin.Long.fromBits(bufInt32[lowIndex], bufInt32[highIndex]);\n};\n\nKotlin.doubleFromBits = function(value) {\n    bufInt32[lowIndex] = value.low_;\n    bufInt32[highIndex] = value.high_;\n    return bufFloat64[0];\n};\n\nKotlin.floatToBits = function(value) {\n    return Kotlin.floatToRawBits(isNaN(value) ? NaN : value);\n};\n\nKotlin.floatToRawBits = function(value) {\n    bufFloat32[0] = value;\n    return bufInt32[0];\n};\n\nKotlin.floatFromBits = function(value) {\n    bufInt32[0] = value;\n    return bufFloat32[0];\n};\n\n// returns zero value for number with positive sign bit and non-zero value for number with negative sign bit.\nKotlin.doubleSignBit = function(value) {\n    bufFloat64[0] = value;\n    return bufInt32[highIndex] & 0x80000000;\n};\n\nKotlin.numberHashCode = function(obj) {\n    if ((obj | 0) === obj) {\n        return obj | 0;\n    }\n    else {\n        bufFloat64[0] = obj;\n        return (bufInt32[highIndex] * 31 | 0) + bufInt32[lowIndex] | 0;\n    }\n}\n\nKotlin.ensureNotNull = function(x) {\n    return x != null ? x : Kotlin.throwNPE();\n};\n\n/*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\nif (typeof String.prototype.startsWith === \"undefined\") {\n    Object.defineProperty(String.prototype, \"startsWith\", {\n        value: function (searchString, position) {\n            position = position || 0;\n            return this.lastIndexOf(searchString, position) === position;\n        }\n    });\n}\n\nif (typeof String.prototype.endsWith === \"undefined\") {\n    Object.defineProperty(String.prototype, \"endsWith\", {\n        value: function (searchString, position) {\n            var subjectString = this.toString();\n            if (position

```

```

=== undefined || position > subjectString.length) {\n          position = subjectString.length;\n          }\n    position -= searchString.length;\n    var lastIndex = subjectString.indexOf(searchString, position);\n    return lastIndex !== -1 && lastIndex === position;\n  }\n  });\n}\n\n// ES6 Math polyfills\n\nif (typeof Math.sign ===\n  \"undefined\") {\n  Math.sign = function(x) {\n    x = +x; // convert to a number\n    if (x === 0 ||\n    isNaN(x)) {\n      return Number(x);\n    }\n    return x > 0 ? 1 : -1;\n  };\n}\n\nif (typeof Math.trunc ===\n  \"undefined\") {\n  Math.trunc = function(x) {\n    if (isNaN(x)) {\n      return NaN;\n    }\n    if (x > 0)\n    {\n      return Math.floor(x);\n    }\n    return Math.ceil(x);\n  };\n}\n\n(function() {\n  var epsilon =\n  2.220446049250313E-16;\n  var taylor_2_bound = Math.sqrt(epsilon);\n  var taylor_n_bound =\n  Math.sqrt(taylor_2_bound);\n  var upper_taylor_2_bound = 1/taylor_2_bound;\n  var upper_taylor_n_bound =\n  1/taylor_n_bound;\n  if (typeof Math.sinh === \"undefined\") {\n    Math.sinh = function(x) {\n      if\n      (Math.abs(x) < taylor_n_bound) {\n        var result = x;\n        if (Math.abs(x) > taylor_2_bound) {\n          result += (x * x * x) / 6;\n        }\n        return result;\n      } else {\n        var y =\n        Math.exp(x);\n        var y1 = 1 / y;\n        if (!isFinite(y)) return Math.exp(x - Math.LN2);\n        if\n        (!isFinite(y1)) return -Math.exp(-x - Math.LN2);\n        return (y - y1) / 2;\n      }\n    };\n  }\n  if\n  (typeof Math.cosh === \"undefined\") {\n    Math.cosh = function(x) {\n      var y = Math.exp(x);\n      var\n      y1 = 1 / y;\n      if (!isFinite(y) || !isFinite(y1)) return Math.exp(Math.abs(x) - Math.LN2);\n      return (y +\n      y1) / 2;\n    };\n  }\n  if (typeof Math.tanh === \"undefined\") {\n    Math.tanh = function(x) {\n      if\n      (Math.abs(x) < taylor_n_bound) {\n        var result = x;\n        if (Math.abs(x) > taylor_2_bound) {\n          result -= (x * x * x) / 3;\n        }\n        return result;\n      }\n      else {\n        var a =\n        Math.exp(+x), b = Math.exp(-x);\n        return a === Infinity ? 1 : b === Infinity ? -1 : (a - b) / (a + b);\n      }\n    };\n  }\n  // Inverse hyperbolic function implementations derived from boost special math functions,\n  // Copyright Eric Ford & Hubert Holin 2001.\n  if (typeof Math.asinh === \"undefined\") {\n    var asinh =\n    function(x) {\n      if (x >= +taylor_n_bound)\n      {\n        if (x > upper_taylor_n_bound)\n        {\n          if (x > upper_taylor_2_bound)\n          {\n            // approximation by laurent series in\n            1/x at 0+ order from -1 to 0\n            return Math.log(x) + Math.LN2;\n          }\n          else\n          {\n            // approximation by laurent series in 1/x at 0+ order from -1 to 1\n            return\n            Math.log(x * 2 + (1 / (x * 2)));\n          }\n        }\n        else if (x <= -taylor_n_bound)\n        {\n          return -asinh(-x);\n        }\n        else\n        {\n          // approximation by taylor series in x at 0 up to\n          order 2\n          var result = x;\n          if (Math.abs(x) >= taylor_2_bound)\n          {\n            var x3 =\n            x * x * x;\n            // approximation by taylor series in x at 0 up to order 4\n            result -= x3 / 6;\n          }\n          return result;\n        }\n      }\n    };\n    Math.asinh = asinh;\n  }\n  if (typeof Math.acosh ===\n  \"undefined\") {\n    Math.acosh = function(x) {\n      if (x < 1)\n      {\n        return NaN;\n      }\n      else if (x - 1 >= taylor_n_bound)\n      {\n        if (x > upper_taylor_2_bound)\n        {\n          // approximation by laurent series in 1/x at 0+ order from -1 to 0\n          return Math.log(x) + Math.LN2;\n        }\n        else\n        {\n          return Math.log(x + Math.sqrt(x * x - 1));\n        }\n      }\n      else\n      {\n        var y = Math.sqrt(x - 1);\n        // approximation by taylor series in y at 0\n        up to order 2\n        var result = y;\n        if (y >= taylor_2_bound)\n        {\n          var y3 = y *\n          y * y;\n          // approximation by taylor series in y at 0 up to order 4\n          result -= y3 / 12;\n        }\n        return Math.sqrt(2) * result;\n      }\n    };\n  }\n  if (typeof Math.atanh === \"undefined\")\n  {\n    Math.atanh = function(x) {\n      if (Math.abs(x) < taylor_n_bound) {\n        var result = x;\n        if (Math.abs(x) > taylor_2_bound) {\n          result += (x * x * x) / 3;\n        }\n        return result;\n      }\n      return Math.log((1 + x) / (1 - x)) / 2;\n    };\n  }\n  if (typeof Math.log1p === \"undefined\") {\n    Math.log1p = function(x) {\n      if (Math.abs(x) < taylor_n_bound) {\n        var x2 = x * x;\n        var x3 = x2 * x;\n        var x4 = x3 * x;\n        // approximation by taylor series in x at 0 up to order 4\n        return (-x4 / 4 + x3 / 3 - x2 / 2 + x);\n      }\n      return Math.log(x + 1);\n    };\n  }\n  if (typeof\n  Math.expm1 === \"undefined\") {\n    Math.expm1 = function(x) {\n      if (Math.abs(x) < taylor_n_bound)\n      {\n        var x2 = x * x;\n        var x3 = x2 * x;\n        var x4 = x3 * x;\n        // approximation by

```

```

taylor series in x at 0 up to order 4\n      return (x4 / 24 + x3 / 6 + x2 / 2 + x);\n      }\n      return
Math.exp(x) - 1;\n    });\n  }\n});\n\nif (typeof Math.hypot === \"undefined\") {\n  Math.hypot = function() {\n    var y = 0;\n    var length = arguments.length;\n    for (var i = 0; i < length; i++) {\n      if (arguments[i]
=== Infinity || arguments[i] === -Infinity) {\n        return Infinity;\n      }\n      y += arguments[i] *
arguments[i];\n    }\n    return Math.sqrt(y);\n  });\n}\n\nif (typeof Math.log10 === \"undefined\") {\n
Math.log10 = function(x) {\n  return Math.log(x) * Math.LOG10E;\n};\n}\n\nif (typeof Math.log2 ===
\"undefined\") {\n  Math.log2 = function(x) {\n    return Math.log(x) * Math.LOG2E;\n };\n}\n\nif (typeof
Math.clz32 === \"undefined\") {\n  Math.clz32 = (function(log, LN2) {\n    return function(x) {\n      var
asUint = x >>> 0;\n      if (asUint === 0) {\n        return 32;\n      }\n      return 31 - (log(asUint) /
LN2 | 0) | 0; // the \"| 0\" acts like math.floor\n    }; \n  })(Math.log, Math.LN2);\n}\n\n// For HtmlUnit and
PhantomJs\n\nif (typeof ArrayBuffer.isView === \"undefined\") {\n  ArrayBuffer.isView = function(a) {\n
return a != null && a.__proto__ != null && a.__proto__.__proto__ === Int8Array.prototype.__proto__;\n
};\n}\n\nif (typeof Array.prototype.fill === \"undefined\") {\n  // Polyfill from https://developer.mozilla.org/en-
US/docs/Web/JavaScript/Reference/Global_Objects/Array/fill#Polyfill\n  Object.defineProperty(Array.prototype,
'fill', {\n    value: function (value) {\n      // Steps 1-2.\n      if (this == null) {\n        throw new
TypeError('this is null or not defined');\n      }\n      var O = Object(this);\n      // Steps 3-5.\n      var
len = O.length >>> 0;\n      // Steps 6-7.\n      var start = arguments[1];\n      var relativeStart = start
>> 0;\n      // Step 8.\n      var k = relativeStart < 0 ?\n        Math.max(len + relativeStart, 0) :\n        Math.min(relativeStart, len);\n      // Steps 9-10.\n      var end = arguments[2];\n      var
relativeEnd = end === undefined ?\n        len : end >> 0;\n      // Step 11.\n      var finalValue
= relativeEnd < 0 ?\n        Math.max(len + relativeEnd, 0) :\n        Math.min(relativeEnd,
len);\n      // Step 12.\n      while (k < finalValue) {\n        O[k] = value;\n        k++;\n      }\n      // Step 13.\n      return O;\n    }; \n  });\n}\n\n(function() {\n  function normalizeOffset(offset, length)
{\n    if (offset < 0) return Math.max(0, offset + length);\n    return Math.min(offset, length);\n  }\n  function
typedArraySlice(begin, end) {\n    if (typeof end === \"undefined\") {\n      end = this.length;\n    }\n    begin =
normalizeOffset(begin || 0, this.length);\n    end = Math.max(begin, normalizeOffset(end, this.length));\n    return
new this.constructor(this.subarray(begin, end));\n  }\n  var arrays = [Int8Array, Int16Array,
Uint16Array, Int32Array, Float32Array, Float64Array];\n  for (var i = 0; i < arrays.length; ++i) {\n    var
TypedArray = arrays[i];\n    if (typeof TypedArray.prototype.fill === \"undefined\") {\n      Object.defineProperty(TypedArray.prototype, 'fill', {\n        value: Array.prototype.fill\n      });\n    }\n    if (typeof TypedArray.prototype.slice === \"undefined\") {\n      Object.defineProperty(TypedArray.prototype, 'slice', {\n        value: typedArraySlice\n      });\n    }\n  }\n  // Patch apply to work with TypedArrays
if needed.\n  try {\n    (function() {}).apply(null, new Int32Array(0))\n  } catch (e) {\n    var apply =
Function.prototype.apply;\n    Object.defineProperty(Function.prototype, 'apply', {\n      value: function(self,
array) {\n        return apply.call(this, self, [].slice.call(array));\n      }\n    });\n  }\n  // Patch map to
work with TypedArrays if needed.\n  for (var i = 0; i < arrays.length; ++i) {\n    var TypedArray = arrays[i];\n    if (typeof TypedArray.prototype.map === \"undefined\") {\n      Object.defineProperty(TypedArray.prototype, 'map', {\n        value: function(callback, self) {\n          return [].slice.call(this).map(callback, self);\n        }\n      });\n    }\n  }\n  // Patch sort to work with TypedArrays if needed.\n  // TODO: consider to
remove following function and replace it with `Kotlin.doubleCompareTo` (see misc.js)\n  var
totalOrderComparator = function (a, b) {\n    if (a < b) return -1;\n    if (a > b) return 1;\n\n    if (a === b) {\n      if (a !== 0) return 0;\n      var ia = 1 / a;\n      return ia === 1 / b ? 0 : (ia < 0 ? -1 : 1);\n    }\n    return a !== a ? (b !== b ? 0 : 1) : -1\n  }; \n  for (var i = 0; i < arrays.length; ++i) {\n    var TypedArray =
arrays[i];\n    if (typeof TypedArray.prototype.sort === \"undefined\") {\n      Object.defineProperty(TypedArray.prototype, 'sort', {\n        value: function(compareFunction) {\n          return Array.prototype.sort.call(this, compareFunction || totalOrderComparator);\n        }\n      });\n    }\n  }\n})();\n\n\"/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors. \n * Use
of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n

```

```

*\n\nKotlin.Kind = {\n  CLASS: \"class\", \n  INTERFACE: \"interface\", \n  OBJECT:
\"object\"}\n};\n\nKotlin.callGetter = function (thisObject, klass, propertyName) {\n  var propertyDescriptor =
Object.getOwnPropertyDescriptor(klass, propertyName);\n  if (propertyDescriptor != null &&
propertyDescriptor.get != null) {\n    return propertyDescriptor.get.call(thisObject);\n  }\n  propertyDescriptor
= Object.getOwnPropertyDescriptor(thisObject, propertyName);\n  if (propertyDescriptor != null && \"value\" in
propertyDescriptor) {\n    return thisObject[propertyName];\n  }\n  return Kotlin.callGetter(thisObject,
Object.getPrototypeOf(klass), propertyName);\n};\n\nKotlin.callSetter = function (thisObject, klass, propertyName,
value) {\n  var propertyDescriptor = Object.getOwnPropertyDescriptor(klass, propertyName);\n  if
(propertyDescriptor != null && propertyDescriptor.set != null) {\n    propertyDescriptor.set.call(thisObject,
value);\n    return;\n  }\n  propertyDescriptor = Object.getOwnPropertyDescriptor(thisObject,
propertyName);\n  if (propertyDescriptor != null && \"value\" in propertyDescriptor) {\n
thisObject[propertyName] = value;\n    return\n  }\n  Kotlin.callSetter(thisObject,
Object.getPrototypeOf(klass), propertyName, value);\n};\n\nfunction isInheritanceFromInterface(ctor, iface) {\n  if
(ctor === iface) return true;\n  var metadata = ctor.$metadata$;\n  if (metadata != null) {\n    var interfaces =
metadata.interfaces;\n    for (var i = 0; i < interfaces.length; i++) {\n      if
(isInheritanceFromInterface(interfaces[i], iface)) {\n        return true;\n      }\n    }\n  }\n  var
superPrototype = ctor.prototype != null ? Object.getPrototypeOf(ctor.prototype) : null;\n  var superConstructor =
superPrototype != null ? superPrototype.constructor : null;\n  return superConstructor != null &&
isInheritanceFromInterface(superConstructor, iface);\n}\n\n/**\n * @param {*} object\n * @param
{Function|Object} klass\n * @returns {Boolean}\n */\nKotlin.isType = function (object, klass) {\n  if (klass ===
Object) {\n    switch (typeof object) {\n      case \"string\":\n      case \"number\":\n      case
\"boolean\":\n      case \"function\":\n        return true;\n      default:\n        return object instanceof
Object;\n    }\n  }\n  if (object == null || klass == null || (typeof object !== 'object' && typeof object !==
'function')) {\n    return false;\n  }\n  if (typeof klass === \"function\" && object instanceof klass) {\n
return true;\n  }\n  var proto = Object.getPrototypeOf(klass);\n  var constructor = proto != null ?
proto.constructor : null;\n  if (constructor != null && \"$metadata$\" in constructor) {\n    var metadata =
constructor.$metadata$;\n    if (metadata.kind === Kotlin.Kind.OBJECT) {\n      return object === klass;\n    }\n  }\n  var klassMetadata = klass.$metadata$;\n  // In WebKit (JavaScriptCore) for some interfaces from
DOM typeof returns \"object\", nevertheless they can be used in RHS of instanceof\n  if (klassMetadata == null)
{\n    return object instanceof klass;\n  }\n  if (klassMetadata.kind === Kotlin.Kind.INTERFACE &&
object.constructor != null) {\n    return isInheritanceFromInterface(object.constructor, klass);\n  }\n  return
false;\n};\n\nKotlin.isNumber = function (a) {\n  return typeof a == \"number\" || a instanceof
Kotlin.Long;\n};\n\nKotlin.isChar = function (value) {\n  return value instanceof
Kotlin.BoxedChar;\n};\n\nKotlin.isComparable = function (value) {\n  var type = typeof value;\n  return type
=== \"string\" ||\n    type === \"boolean\" ||\n    Kotlin.isNumber(value) ||\n    Kotlin.isType(value,
Kotlin.kotlin.Comparable);\n};\n\nKotlin.isCharSequence = function (value) {\n  return typeof value === \"string\"
|| Kotlin.isType(value, Kotlin.kotlin.CharSequence);\n};\n\n/*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin
Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.\n */\n\n// a package is omitted to get declarations directly under the
module\n\n@PublishedApi\nexternal fun <T> Array(size: Int): Array<T>\n\n@JsName(\"newArray\")\nfun
<T> newArray(size: Int, initialValue: T) = fillArrayVal(Array<T>(size),
initialValue)\n\n@JsName(\"newArrayF\")\ninline fun <T> arrayWithFun(size: Int, init: (Int) -> T) =
fillArrayFun(Array<T>(size), init)\n\n@JsName(\"fillArray\")\ninline fun <T> fillArrayFun(array: Array<T>,
init: (Int) -> T): Array<T> {\n  for (i in 0..array.size - 1) {\n    array[i] = init(i)\n  }\n  return
array\n}\n\n@JsName(\"booleanArray\")\nfun booleanArray(size: Int, init: dynamic): Array<Boolean> {\n  val
result: dynamic = Array<Boolean>(size)\n  result.$type$ = \"BooleanArray\"\n  return when (init) {\n    null,
true -> fillArrayVal(result, false)\n    false -> result\n    else -> fillArrayFun<Boolean>(result, init)\n  }\n}\n\n@JsName(\"booleanArrayF\")\ninline fun booleanArrayWithFun(size: Int, init: (Int) -> Boolean):

```

```

Array<Boolean> = fillArrayFun(booleanArray(size, false),
init)\n\n@JsName("charArray")\n@Suppress("UNUSED_PARAMETER")\nfun charArray(size: Int, init:
dynamic): Array<Char> {\n    val result = js("new Uint16Array(size)")\n    result.`$type$` = "CharArray"\n
return when (init) {\n        null, true, false -> result // For consistency\n        else -> fillArrayFun<Char>(result,
init)\n    }\n}\n\n@JsName("charArrayF")\ninline fun charArrayWithFun(size: Int, init: (Int) -> Char):
Array<Char> {\n    val array = charArray(size, null)\n    for (i in 0..array.size - 1) {\n
@Suppress("UNUSED_VARIABLE") // used in js block\n        val value = init(i)\n        js("array[i] = value;")\n
}\n    return array\n}\n\n@JsName("untypedCharArrayF")\ninline fun untypedCharArrayWithFun(size: Int, init:
(Int) -> Char): Array<Char> {\n    val array = Array<Char>(size)\n    for (i in 0..array.size - 1) {\n
@Suppress("UNUSED_VARIABLE") // used in js block\n        val value = init(i)\n        js("array[i] = value;")\n
}\n    return array\n}\n\n@JsName("longArray")\nfun longArray(size: Int, init: dynamic): Array<Long> {\n    val
result: dynamic = Array<Long>(size)\n    result.`$type$` = "LongArray"\n    return when (init) {\n        null, true ->
fillArrayVal(result, 0L)\n        false -> result\n        else -> fillArrayFun<Long>(result, init)\n
}\n}\n\n@JsName("longArrayF")\ninline fun longArrayWithFun(size: Int, init: (Int) -> Long): Array<Long> =
fillArrayFun(longArray(size, false), init)\n\nprivate fun <T> fillArrayVal(array: Array<T>, initialValue: T): Array<T>
{\n    for (i in 0..array.size - 1) {\n        array[i] = initialValue\n    }\n    return array\n}", "/*\n * Copyright 2010-2018
JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin\n\npublic class
Enum<T : Enum<T>> : Comparable<Enum<T>> {\n    @JsName("name$") private var _name: String = ""\n
@JsName("ordinal$") private var _ordinal: Int = 0\n    val name: String\n        get() = _name\n    val ordinal:
Int\n        get() = _ordinal\n    override fun compareTo(other: Enum<T>) = ordinal.compareTo(other.ordinal)\n
override fun equals(other: Any?) = this === other\n    override fun hashCode(): Int =
js("Kotlin.identityHashCode")(this)\n    override fun toString() = name\n    companion object\n}", "/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is
governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage
kotlin.js.internal\n\n@JsName("DoubleCompanionObject")\ninternal object DoubleCompanionObject {\n
@JsName("MIN_VALUE")\n    const val MIN_VALUE: Double = 4.9E-324\n
@JsName("MAX_VALUE")\n    const val MAX_VALUE: Double = 1.7976931348623157E308\n
@JsName("POSITIVE_INFINITY")\n    @Suppress("DIVISION_BY_ZERO")\n    const val
POSITIVE_INFINITY: Double = 1.0 / 0.0\n
@JsName("NEGATIVE_INFINITY")\n    @Suppress("DIVISION_BY_ZERO")\n    const val NEGATIVE_INFINITY: Double = -1.0 / 0.0\n
@JsName("NaN")\n    @Suppress("DIVISION_BY_ZERO")\n    const val NaN: Double = -(0.0 / 0.0)\n
@JsName("SIZE_BYTES")\n    const val SIZE_BYTES = 8\n
@JsName("SIZE_BITS")\n    const val
SIZE_BITS = 64\n}\n\n@JsName("FloatCompanionObject")\ninternal object FloatCompanionObject {\n
@JsName("MIN_VALUE")\n    const val MIN_VALUE: Float = 1.4E-45F\n
@JsName("MAX_VALUE")\n    const val MAX_VALUE: Float = 3.4028235E38F\n
@JsName("POSITIVE_INFINITY")\n    @Suppress("DIVISION_BY_ZERO")\n    const val POSITIVE_INFINITY: Float = 1.0F / 0.0F\n
@JsName("NEGATIVE_INFINITY")\n    @Suppress("DIVISION_BY_ZERO")\n    const val
NEGATIVE_INFINITY: Float = -1.0F / 0.0F\n
@JsName("NaN")\n    @Suppress("DIVISION_BY_ZERO")\n    const val NaN: Float = -(0.0F / 0.0F)\n
@JsName("SIZE_BYTES")\n    const val SIZE_BYTES = 4\n
@JsName("SIZE_BITS")\n    const val
SIZE_BITS = 32\n}\n\n@JsName("IntCompanionObject")\ninternal object IntCompanionObject {\n
@JsName("MIN_VALUE")\n    val MIN_VALUE: Int = -2147483647 - 1\n
@JsName("MAX_VALUE")\n    val MAX_VALUE: Int = 2147483647\n
@JsName("SIZE_BYTES")\n    const val SIZE_BYTES = 4\n
@JsName("SIZE_BITS")\n    const val SIZE_BITS = 32\n}\n\n@JsName("LongCompanionObject")\ninternal
object LongCompanionObject {\n    @JsName("MIN_VALUE")\n    val MIN_VALUE: Long =
js("Kotlin.Long.MIN_VALUE")\n
@JsName("MAX_VALUE")\n    val MAX_VALUE: Long =
js("Kotlin.Long.MAX_VALUE")\n
@JsName("SIZE_BYTES")\n    const val SIZE_BYTES = 8\n

```

```

@JsName("SIZE_BITS")\n const val SIZE_BITS = 64\n}\n\n@JsName("ShortCompanionObject")\ninternal
object ShortCompanionObject {\n @JsName("MIN_VALUE")\n val MIN_VALUE: Short = -32768\n\n
@JsName("MAX_VALUE")\n val MAX_VALUE: Short = 32767\n\n @JsName("SIZE_BYTES")\n const
val SIZE_BYTES = 2\n\n @JsName("SIZE_BITS")\n const val SIZE_BITS =
16\n}\n\n@JsName("ByteCompanionObject")\ninternal object ByteCompanionObject {\n
@JsName("MIN_VALUE")\n val MIN_VALUE: Byte = -128\n\n @JsName("MAX_VALUE")\n val
MAX_VALUE: Byte = 127\n\n @JsName("SIZE_BYTES")\n const val SIZE_BYTES = 1\n\n
@JsName("SIZE_BITS")\n const val SIZE_BITS = 8\n}\n\n@JsName("CharCompanionObject")\ninternal
object CharCompanionObject {\n @JsName("MIN_VALUE")\n public const val MIN_VALUE: Char =
"\u0000"\n\n @JsName("MAX_VALUE")\n public const val MAX_VALUE: Char = "\uFFFF"\n\n
@JsName("MIN_HIGH_SURROGATE")\n public const val MIN_HIGH_SURROGATE: Char = "\uD800"\n\n
@JsName("MAX_HIGH_SURROGATE")\n public const val MAX_HIGH_SURROGATE: Char =
"\uDBFF"\n\n @JsName("MIN_LOW_SURROGATE")\n public const val MIN_LOW_SURROGATE: Char =
"\uDC00"\n\n @JsName("MAX_LOW_SURROGATE")\n public const val MAX_LOW_SURROGATE: Char
= "\uDFFF"\n\n @JsName("MIN_SURROGATE")\n public const val MIN_SURROGATE: Char =
MIN_HIGH_SURROGATE\n\n @JsName("MAX_SURROGATE")\n public const val MAX_SURROGATE:
Char = MAX_LOW_SURROGATE\n\n @JsName("SIZE_BYTES")\n const val SIZE_BYTES = 2\n\n
@JsName("SIZE_BITS")\n const val SIZE_BITS = 16\n}\n\ninternal object StringCompanionObject
{\n}\n\ninternal object BooleanCompanionObject {\n}\n\n"/\n\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin
Programming Language contributors.\n\n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.\n\n
*/\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("ArraysKt")\n\npackage
kotlin.collections\n\n/\n\n// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt\n// See:
https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\n/\n\n\nimport kotlin.random.*\nimport
kotlin.ranges.contains\nimport kotlin.ranges.reversed\n\n\n/**\n * Returns 1st *element* from the array.\n * \n * If the
size of this array is less than 1, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior
is unspecified.\n */\n\n@kotlin.internal.InlineOnly\npublic inline operator fun <T> Array<out T>.component1(): T
{\n return get(0)\n}\n\n/**\n * Returns 1st *element* from the array.\n * \n * If the size of this array is less than 1,
throws an [IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n */\n\n@kotlin.internal.InlineOnly\npublic inline operator fun ByteArray.component1(): Byte {\n return
get(0)\n}\n\n/**\n * Returns 1st *element* from the array.\n * \n * If the size of this array is less than 1, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n */\n\n@kotlin.internal.InlineOnly\npublic inline operator fun ShortArray.component1(): Short {\n return
get(0)\n}\n\n/**\n * Returns 1st *element* from the array.\n * \n * If the size of this array is less than 1, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n */\n\n@kotlin.internal.InlineOnly\npublic inline operator fun IntArray.component1(): Int {\n return
get(0)\n}\n\n/**\n * Returns 1st *element* from the array.\n * \n * If the size of this array is less than 1, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n */\n\n@kotlin.internal.InlineOnly\npublic inline operator fun LongArray.component1(): Long {\n return
get(0)\n}\n\n/**\n * Returns 1st *element* from the array.\n * \n * If the size of this array is less than 1, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n */\n\n@kotlin.internal.InlineOnly\npublic inline operator fun FloatArray.component1(): Float {\n return
get(0)\n}\n\n/**\n * Returns 1st *element* from the array.\n * \n * If the size of this array is less than 1, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n */\n\n@kotlin.internal.InlineOnly\npublic inline operator fun DoubleArray.component1(): Double {\n return
get(0)\n}\n\n/**\n * Returns 1st *element* from the array.\n * \n * If the size of this array is less than 1, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n */\n\n@kotlin.internal.InlineOnly\npublic inline operator fun BooleanArray.component1(): Boolean {\n return

```

`get(0)`\n\n\*\*\n \* Returns 1st \*element\* from the array.\n \* \n \* If the size of this array is less than 1, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun CharArray.component1(): Char {\n return`

`get(0)`\n\n\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \* If the size of this array is less than 2, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun <T> Array<out T>.component2(): T {\n return`

`get(1)`\n\n\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \* If the size of this array is less than 2, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun ByteArray.component2(): Byte {\n return`

`get(1)`\n\n\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \* If the size of this array is less than 2, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun ShortArray.component2(): Short {\n return`

`get(1)`\n\n\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \* If the size of this array is less than 2, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun IntArray.component2(): Int {\n return`

`get(1)`\n\n\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \* If the size of this array is less than 2, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun LongArray.component2(): Long {\n return`

`get(1)`\n\n\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \* If the size of this array is less than 2, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun FloatArray.component2(): Float {\n return`

`get(1)`\n\n\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \* If the size of this array is less than 2, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun DoubleArray.component2(): Double {\n return`

`get(1)`\n\n\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \* If the size of this array is less than 2, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun BooleanArray.component2(): Boolean {\n return`

`get(1)`\n\n\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \* If the size of this array is less than 2, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun CharArray.component2(): Char {\n return`

`get(1)`\n\n\*\*\n \* Returns 3rd \*element\* from the array.\n \* \n \* If the size of this array is less than 3, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun <T> Array<out T>.component3(): T {\n return`

`get(2)`\n\n\*\*\n \* Returns 3rd \*element\* from the array.\n \* \n \* If the size of this array is less than 3, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun ByteArray.component3(): Byte {\n return`

`get(2)`\n\n\*\*\n \* Returns 3rd \*element\* from the array.\n \* \n \* If the size of this array is less than 3, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun ShortArray.component3(): Short {\n return`

`get(2)`\n\n\*\*\n \* Returns 3rd \*element\* from the array.\n \* \n \* If the size of this array is less than 3, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun IntArray.component3(): Int {\n return`

`get(2)`\n\n\*\*\n \* Returns 3rd \*element\* from the array.\n \* \n \* If the size of this array is less than 3, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun LongArray.component3(): Long {\n return`

`get(2)`\n\n\*\*\n \* Returns 3rd \*element\* from the array.\n \* \n \* If the size of this array is less than 3, throws an `[IndexOutOfBoundsException]` except in Kotlin/JS\n \* where the behavior is unspecified.\n

`*\n@kotlin.internal.InlineOnly\npublic inline operator fun FloatArray.component3(): Float {\n return`

```

get(2)\n}\n\n/**\n * Returns 3rd *element* from the array.\n * \n * If the size of this array is less than 3, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun DoubleArray.component3(): Double {\n    return
get(2)\n}\n\n/**\n * Returns 3rd *element* from the array.\n * \n * If the size of this array is less than 3, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun BooleanArray.component3(): Boolean {\n    return
get(2)\n}\n\n/**\n * Returns 3rd *element* from the array.\n * \n * If the size of this array is less than 3, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun CharArray.component3(): Char {\n    return
get(2)\n}\n\n/**\n * Returns 4th *element* from the array.\n * \n * If the size of this array is less than 4, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun <T> Array<out T>.component4(): T {\n    return
get(3)\n}\n\n/**\n * Returns 4th *element* from the array.\n * \n * If the size of this array is less than 4, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun ByteArray.component4(): Byte {\n    return
get(3)\n}\n\n/**\n * Returns 4th *element* from the array.\n * \n * If the size of this array is less than 4, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun ShortArray.component4(): Short {\n    return
get(3)\n}\n\n/**\n * Returns 4th *element* from the array.\n * \n * If the size of this array is less than 4, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun IntArray.component4(): Int {\n    return
get(3)\n}\n\n/**\n * Returns 4th *element* from the array.\n * \n * If the size of this array is less than 4, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun LongArray.component4(): Long {\n    return
get(3)\n}\n\n/**\n * Returns 4th *element* from the array.\n * \n * If the size of this array is less than 4, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun FloatArray.component4(): Float {\n    return
get(3)\n}\n\n/**\n * Returns 4th *element* from the array.\n * \n * If the size of this array is less than 4, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun DoubleArray.component4(): Double {\n    return
get(3)\n}\n\n/**\n * Returns 4th *element* from the array.\n * \n * If the size of this array is less than 4, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun BooleanArray.component4(): Boolean {\n    return
get(3)\n}\n\n/**\n * Returns 4th *element* from the array.\n * \n * If the size of this array is less than 4, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun CharArray.component4(): Char {\n    return
get(3)\n}\n\n/**\n * Returns 5th *element* from the array.\n * \n * If the size of this array is less than 5, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun <T> Array<out T>.component5(): T {\n    return
get(4)\n}\n\n/**\n * Returns 5th *element* from the array.\n * \n * If the size of this array is less than 5, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun ByteArray.component5(): Byte {\n    return
get(4)\n}\n\n/**\n * Returns 5th *element* from the array.\n * \n * If the size of this array is less than 5, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun ShortArray.component5(): Short {\n    return
get(4)\n}\n\n/**\n * Returns 5th *element* from the array.\n * \n * If the size of this array is less than 5, throws an
[IndexOutOfBoundsException] except in Kotlin/JS\n * where the behavior is unspecified.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun IntArray.component5(): Int {\n    return

```

`get(4)` Returns 5th element from the array. If the size of this array is less than 5, throws an `IndexOutOfBoundsException` except in Kotlin/JS where the behavior is unspecified.

```

@kotlin.internal.InlineOnly
public inline operator fun LongArray.component5(): Long {
    return get(4)
}

```

`get(4)` Returns 5th element from the array. If the size of this array is less than 5, throws an `IndexOutOfBoundsException` except in Kotlin/JS where the behavior is unspecified.

```

@kotlin.internal.InlineOnly
public inline operator fun FloatArray.component5(): Float {
    return get(4)
}

```

`get(4)` Returns 5th element from the array. If the size of this array is less than 5, throws an `IndexOutOfBoundsException` except in Kotlin/JS where the behavior is unspecified.

```

@kotlin.internal.InlineOnly
public inline operator fun DoubleArray.component5(): Double {
    return get(4)
}

```

`get(4)` Returns 5th element from the array. If the size of this array is less than 5, throws an `IndexOutOfBoundsException` except in Kotlin/JS where the behavior is unspecified.

```

@kotlin.internal.InlineOnly
public inline operator fun BooleanArray.component5(): Boolean {
    return get(4)
}

```

`get(4)` Returns 5th element from the array. If the size of this array is less than 5, throws an `IndexOutOfBoundsException` except in Kotlin/JS where the behavior is unspecified.

```

@kotlin.internal.InlineOnly
public inline operator fun CharArray.component5(): Char {
    return get(4)
}

```

`contains(element)` Returns true if [element] is found in the array.

```

@kotlin.internal.OnlyInputTypes T
Array<out T>.contains(element: T): Boolean {
    return indexOf(element) >= 0
}

```

`contains(element)` Returns true if [element] is found in the array.

```

public operator fun
ByteArray.contains(element: Byte): Boolean {
    return indexOf(element) >= 0
}

```

`contains(element)` Returns true if [element] is found in the array.

```

public operator fun
ShortArray.contains(element: Short): Boolean {
    return indexOf(element) >= 0
}

```

`contains(element)` Returns true if [element] is found in the array.

```

public operator fun
IntArray.contains(element: Int): Boolean {
    return indexOf(element) >= 0
}

```

`contains(element)` Returns true if [element] is found in the array.

```

@Deprecated("The function has unclear behavior when searching for NaN or zero values and will be removed soon. Use 'any { it == element }' instead to continue using this behavior, or '.asList().contains(element: T)' to get the same search behavior as in a list.", ReplaceWith("any { it == element }"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.6")
@Suppress("DEPRECATION_ERROR")
public operator fun
FloatArray.contains(element: Float): Boolean {
    return indexOf(element) >= 0
}

```

`contains(element)` Returns true if [element] is found in the array.

```

@Deprecated("The function has unclear behavior when searching for NaN or zero values and will be removed soon. Use 'any { it == element }' instead to continue using this behavior, or '.asList().contains(element: T)' to get the same search behavior as in a list.", ReplaceWith("any { it == element }"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.6")
@Suppress("DEPRECATION_ERROR")
public operator fun
DoubleArray.contains(element: Double): Boolean {
    return indexOf(element) >= 0
}

```

`contains(element)` Returns true if [element] is found in the array.

```

public operator fun
BooleanArray.contains(element: Boolean): Boolean {
    return indexOf(element) >= 0
}

```

`contains(element)` Returns true if [element] is found in the array.

```

public operator fun
CharArray.contains(element: Char): Boolean {
    return indexOf(element) >= 0
}

```

`elementAt(index)` Returns an element at the given [index] or throws an `IndexOutOfBoundsException` if the [index] is out of bounds of this array.

```

@sample samples.collections.Collections.Elements.elementAt
public expect fun <T> Array<out T>.elementAt(index: Int): T

```

`elementAt(index)` Returns an element at the given [index] or throws an `IndexOutOfBoundsException` if the [index] is out of bounds of this array.

```

@sample samples.collections.Collections.Elements.elementAt
public expect fun
ByteArray.elementAt(index: Int): Byte

```

`elementAt(index)` Returns an element at the given [index] or throws an `IndexOutOfBoundsException` if the [index] is out of bounds of this array.

```

@sample samples.collections.Collections.Elements.elementAt
public expect fun
ShortArray.elementAt(index: Int): Short

```

`elementAt(index)` Returns an element at the given [index] or throws an `IndexOutOfBoundsException` if the [index] is out of bounds of this array.

```

@sample samples.collections.Collections.Elements.elementAt
public expect fun
IntArray.elementAt(index: Int): Int

```

`Int`\n\n\*\*\n \* Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAt\n \* \n\npublic expect fun LongArray.elementAt(index: Int): Long\n\n\*\*\n \* Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAt\n \* \n\npublic expect fun FloatArray.elementAt(index: Int): Float\n\n\*\*\n \* Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAt\n \* \n\npublic expect fun DoubleArray.elementAt(index: Int): Double\n\n\*\*\n \* Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAt\n \* \n\npublic expect fun BooleanArray.elementAt(index: Int): Boolean\n\n\*\*\n \* Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAt\n \* \n\npublic expect fun CharArray.elementAt(index: Int): Char\n\n\*\*\n \* Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAtOrElse\n \* \n\n@kotlin.internal.InlineOnly\n\npublic inline fun <T> Array<out T>.elementAtOrElse(index: Int, defaultValue: (Int) -> T): T {\n return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n\*\*\n \* Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAtOrElse\n \* \n\n@kotlin.internal.InlineOnly\n\npublic inline fun ByteArray.elementAtOrElse(index: Int, defaultValue: (Int) -> Byte): Byte {\n return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n\*\*\n \* Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAtOrElse\n \* \n\n@kotlin.internal.InlineOnly\n\npublic inline fun ShortArray.elementAtOrElse(index: Int, defaultValue: (Int) -> Short): Short {\n return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n\*\*\n \* Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAtOrElse\n \* \n\n@kotlin.internal.InlineOnly\n\npublic inline fun IntArray.elementAtOrElse(index: Int, defaultValue: (Int) -> Int): Int {\n return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n\*\*\n \* Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAtOrElse\n \* \n\n@kotlin.internal.InlineOnly\n\npublic inline fun LongArray.elementAtOrElse(index: Int, defaultValue: (Int) -> Long): Long {\n return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n\*\*\n \* Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAtOrElse\n \* \n\n@kotlin.internal.InlineOnly\n\npublic inline fun FloatArray.elementAtOrElse(index: Int, defaultValue: (Int) -> Float): Float {\n return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n\*\*\n \* Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAtOrElse\n \* \n\n@kotlin.internal.InlineOnly\n\npublic inline fun DoubleArray.elementAtOrElse(index: Int, defaultValue: (Int) -> Double): Double {\n return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n\*\*\n \* Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAtOrElse\n \* \n\n@kotlin.internal.InlineOnly\n\npublic inline fun BooleanArray.elementAtOrElse(index: Int, defaultValue: (Int) -> Boolean): Boolean {\n return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n\*\*\n \* Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n \* \n \* @sample samples.collections.Collections.Elements.elementAtOrElse\n \* \n\n@kotlin.internal.InlineOnly\n\npublic inline fun CharArray.elementAtOrElse(index: Int, defaultValue: (Int) -> Char): Char {\n return if (index >= 0 && index <=

lastIndex) get(index) else defaultValue(index)\n}\n\n/\*\*\n \* Returns an element at the given [index] or `null` if the [index] is out of bounds of this array.\n \* \n \* @sample

```

samples.collections.Collections.Elements.elementAtOrNull\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
<T> Array<out T>.elementAtOrNull(index: Int): T? {\n  return this.getOrNull(index)\n}\n\n/**\n * Returns an
element at the given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
ByteArray.elementAtOrNull(index: Int): Byte? {\n  return this.getOrNull(index)\n}\n\n/**\n * Returns an element
at the given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
ShortArray.elementAtOrNull(index: Int): Short? {\n  return this.getOrNull(index)\n}\n\n/**\n * Returns an element
at the given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
IntArray.elementAtOrNull(index: Int): Int? {\n  return this.getOrNull(index)\n}\n\n/**\n * Returns an element at
the given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
LongArray.elementAtOrNull(index: Int): Long? {\n  return this.getOrNull(index)\n}\n\n/**\n * Returns an element
at the given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
FloatArray.elementAtOrNull(index: Int): Float? {\n  return this.getOrNull(index)\n}\n\n/**\n * Returns an element
at the given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
DoubleArray.elementAtOrNull(index: Int): Double? {\n  return this.getOrNull(index)\n}\n\n/**\n * Returns an
element at the given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
BooleanArray.elementAtOrNull(index: Int): Boolean? {\n  return this.getOrNull(index)\n}\n\n/**\n * Returns an
element at the given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
CharArray.elementAtOrNull(index: Int): Char? {\n  return this.getOrNull(index)\n}\n\n/**\n * Returns the first
element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n *^\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out
T>.find(predicate: (T) -> Boolean): T? {\n  return firstOrNull(predicate)\n}\n\n/**\n * Returns the first element
matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
ByteArray.find(predicate: (Byte) -> Boolean): Byte? {\n  return firstOrNull(predicate)\n}\n\n/**\n * Returns the
first element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
ShortArray.find(predicate: (Short) -> Boolean): Short? {\n  return firstOrNull(predicate)\n}\n\n/**\n * Returns the
first element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
IntArray.find(predicate: (Int) -> Boolean): Int? {\n  return firstOrNull(predicate)\n}\n\n/**\n * Returns the first
element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
LongArray.find(predicate: (Long) -> Boolean): Long? {\n  return firstOrNull(predicate)\n}\n\n/**\n * Returns the
first element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n *^\n@kotlin.internal.InlineOnly\npublic inline fun
FloatArray.find(predicate: (Float) -> Boolean): Float? {\n  return firstOrNull(predicate)\n}\n\n/**\n * Returns the
first element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n *^\n@kotlin.internal.InlineOnly\npublic inline fun

```

`DoubleArray.find(predicate: (Double) -> Boolean): Double?` `{\n return firstOrNull(predicate)\n}\n\n/**\n * Returns the first element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun`

`BooleanArray.find(predicate: (Boolean) -> Boolean): Boolean?` `{\n return firstOrNull(predicate)\n}\n\n/**\n * Returns the first element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun`

`CharArray.find(predicate: (Char) -> Boolean): Char?` `{\n return firstOrNull(predicate)\n}\n\n/**\n * Returns the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun`

`<T> Array<out T>.findLast(predicate: (T) -> Boolean): T?` `{\n return lastOrNull(predicate)\n}\n\n/**\n * Returns the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun`

`ByteArray.findLast(predicate: (Byte) -> Boolean): Byte?` `{\n return lastOrNull(predicate)\n}\n\n/**\n * Returns the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun`

`ShortArray.findLast(predicate: (Short) -> Boolean): Short?` `{\n return lastOrNull(predicate)\n}\n\n/**\n * Returns the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun`

`IntArray.findLast(predicate: (Int) -> Boolean): Int?` `{\n return lastOrNull(predicate)\n}\n\n/**\n * Returns the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun`

`LongArray.findLast(predicate: (Long) -> Boolean): Long?` `{\n return lastOrNull(predicate)\n}\n\n/**\n * Returns the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun`

`FloatArray.findLast(predicate: (Float) -> Boolean): Float?` `{\n return lastOrNull(predicate)\n}\n\n/**\n * Returns the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun`

`DoubleArray.findLast(predicate: (Double) -> Boolean): Double?` `{\n return lastOrNull(predicate)\n}\n\n/**\n * Returns the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun`

`BooleanArray.findLast(predicate: (Boolean) -> Boolean): Boolean?` `{\n return lastOrNull(predicate)\n}\n\n/**\n * Returns the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun`

`CharArray.findLast(predicate: (Char) -> Boolean): Char?` `{\n return lastOrNull(predicate)\n}\n\n/**\n * Returns first element.\n * @throws [NoSuchElementException] if the array is empty.\n */\n@kotlin.internal.InlineOnly\npublic fun`

`<T> Array<out T>.first(): T` `{\n if (isEmpty())\n throw NoSuchElementException("Array is empty.")\n return this[0]\n}\n\n/**\n * Returns first element.\n * @throws [NoSuchElementException] if the array is empty.\n */\n@kotlin.internal.InlineOnly\npublic fun`

`ByteArray.first(): Byte` `{\n if (isEmpty())\n throw NoSuchElementException("Array is empty.")\n return this[0]\n}\n\n/**\n * Returns first element.\n * @throws [NoSuchElementException] if the array is empty.\n */\n@kotlin.internal.InlineOnly\npublic fun`

`ShortArray.first(): Short` `{\n if (isEmpty())\n throw NoSuchElementException("Array is empty.")\n return this[0]\n}\n\n/**\n * Returns first element.\n * @throws [NoSuchElementException] if the array is empty.\n */\n@kotlin.internal.InlineOnly\npublic fun`

`IntArray.first(): Int` `{\n if (isEmpty())\n throw NoSuchElementException("Array is empty.")\n return this[0]\n}\n\n/**\n * Returns first element.\n * @throws [NoSuchElementException] if the array is empty.\n */\n@kotlin.internal.InlineOnly\npublic fun`

`LongArray.first(): Long` `{\n if (isEmpty())\n throw NoSuchElementException("Array is empty.")\n return this[0]\n}\n\n/**\n * Returns first element.\n * @throws [NoSuchElementException] if the array is empty.\n */\n@kotlin.internal.InlineOnly\npublic fun`

`FloatArray.first(): Float` `{\n if (isEmpty())\n throw NoSuchElementException("Array is empty.")\n return this[0]\n}\n\n/**\n * Returns first element.\n * @throws [NoSuchElementException] if the array is empty.\n */\n@kotlin.internal.InlineOnly\npublic fun`

```

DoubleArray.first(): Double {
    if (isEmpty())
        throw NoSuchElementException("Array is empty.")
    return this[0]
}
/**
 * Returns first element.
 * @throws [NoSuchElementException] if the array is empty.
 */
public fun BooleanArray.first(): Boolean {
    if (isEmpty())
        throw NoSuchElementException("Array is empty.")
    return this[0]
}
/**
 * Returns first element.
 * @throws [NoSuchElementException] if the array is empty.
 */
public fun CharArray.first(): Char {
    if (isEmpty())
        throw NoSuchElementException("Array is empty.")
    return this[0]
}
/**
 * Returns the first element matching the given [predicate].
 * @throws [NoSuchElementException] if no such element is found.
 */
public inline fun <T> Array<out T>.first(predicate: (T) -> Boolean): T {
    for (element in this)
        if (predicate(element))
            return element
    throw NoSuchElementException("Array contains no element matching the predicate.")
}
/**
 * Returns the first element matching the given [predicate].
 * @throws [NoSuchElementException] if no such element is found.
 */
public inline fun ByteArray.first(predicate: (Byte) -> Boolean): Byte {
    for (element in this)
        if (predicate(element))
            return element
    throw NoSuchElementException("Array contains no element matching the predicate.")
}
/**
 * Returns the first element matching the given [predicate].
 * @throws [NoSuchElementException] if no such element is found.
 */
public inline fun ShortArray.first(predicate: (Short) -> Boolean): Short {
    for (element in this)
        if (predicate(element))
            return element
    throw NoSuchElementException("Array contains no element matching the predicate.")
}
/**
 * Returns the first element matching the given [predicate].
 * @throws [NoSuchElementException] if no such element is found.
 */
public inline fun IntArray.first(predicate: (Int) -> Boolean): Int {
    for (element in this)
        if (predicate(element))
            return element
    throw NoSuchElementException("Array contains no element matching the predicate.")
}
/**
 * Returns the first element matching the given [predicate].
 * @throws [NoSuchElementException] if no such element is found.
 */
public inline fun LongArray.first(predicate: (Long) -> Boolean): Long {
    for (element in this)
        if (predicate(element))
            return element
    throw NoSuchElementException("Array contains no element matching the predicate.")
}
/**
 * Returns the first element matching the given [predicate].
 * @throws [NoSuchElementException] if no such element is found.
 */
public inline fun FloatArray.first(predicate: (Float) -> Boolean): Float {
    for (element in this)
        if (predicate(element))
            return element
    throw NoSuchElementException("Array contains no element matching the predicate.")
}
/**
 * Returns the first element matching the given [predicate].
 * @throws [NoSuchElementException] if no such element is found.
 */
public inline fun DoubleArray.first(predicate: (Double) -> Boolean): Double {
    for (element in this)
        if (predicate(element))
            return element
    throw NoSuchElementException("Array contains no element matching the predicate.")
}
/**
 * Returns the first element matching the given [predicate].
 * @throws [NoSuchElementException] if no such element is found.
 */
public inline fun BooleanArray.first(predicate: (Boolean) -> Boolean): Boolean {
    for (element in this)
        if (predicate(element))
            return element
    throw NoSuchElementException("Array contains no element matching the predicate.")
}
/**
 * Returns the first element matching the given [predicate].
 * @throws [NoSuchElementException] if no such element is found.
 */
public inline fun CharArray.first(predicate: (Char) -> Boolean): Char {
    for (element in this)
        if (predicate(element))
            return element
    throw NoSuchElementException("Array contains no element matching the predicate.")
}
/**
 * Returns the first non-null value produced by [transform] function being applied to elements of this array in iteration order,
 * or throws [NoSuchElementException] if no non-null value was produced.
 */
@sample
samples.collections.Collections.Transformations.firstNotNullOf
/**
 * @SinceKotlin("1.5")
 * @kotlin.internal.InlineOnly
 */
public inline fun <T, R : Any> Array<out T>.firstNotNullOf(transform: (T) -> R?): R {
    return firstNotNullOfOrNull(transform) ?: throw NoSuchElementException("No element of the array was transformed to a non-null value.")
}
/**
 * Returns the first non-null value produced by [transform] function being applied to elements of this array in iteration order,
 * or `null` if no non-null value was produced.
 */
@sample
samples.collections.Collections.Transformations.firstNotNullOf
/**
 * @SinceKotlin("1.5")
 * @kotlin.internal.InlineOnly
 */
public inline fun <T, R : Any> Array<out T>.firstNotNullOfOrNull(transform: (T) -> R?): R? {
    for (element in this) {
        val result =

```

```

transform(element)\n    if (result != null) {\n        return result\n    }\n    return null\n}\n\n**\n *
Returns the first element, or `null` if the array is empty.\n *^/npublic fun <T> Array<out T>.firstOrNull(): T? {\n
return if (isEmpty()) null else this[0]\n}\n\n**\n * Returns the first element, or `null` if the array is empty.\n
*^/npublic fun ByteArray.firstOrNull(): Byte? {\n    return if (isEmpty()) null else this[0]\n}\n\n**\n * Returns the
first element, or `null` if the array is empty.\n *^/npublic fun ShortArray.firstOrNull(): Short? {\n    return if
(isEmpty()) null else this[0]\n}\n\n**\n * Returns the first element, or `null` if the array is empty.\n *^/npublic fun
IntArray.firstOrNull(): Int? {\n    return if (isEmpty()) null else this[0]\n}\n\n**\n * Returns the first element, or
`null` if the array is empty.\n *^/npublic fun LongArray.firstOrNull(): Long? {\n    return if (isEmpty()) null else
this[0]\n}\n\n**\n * Returns the first element, or `null` if the array is empty.\n *^/npublic fun
FloatArray.firstOrNull(): Float? {\n    return if (isEmpty()) null else this[0]\n}\n\n**\n * Returns the first element,
or `null` if the array is empty.\n *^/npublic fun DoubleArray.firstOrNull(): Double? {\n    return if (isEmpty()) null
else this[0]\n}\n\n**\n * Returns the first element, or `null` if the array is empty.\n *^/npublic fun
BooleanArray.firstOrNull(): Boolean? {\n    return if (isEmpty()) null else this[0]\n}\n\n**\n * Returns the first
element, or `null` if the array is empty.\n *^/npublic fun CharArray.firstOrNull(): Char? {\n    return if (isEmpty())
null else this[0]\n}\n\n**\n * Returns the first element matching the given [predicate], or `null` if element was not
found.\n *^/npublic inline fun <T> Array<out T>.firstOrNull(predicate: (T) -> Boolean): T? {\n    for (element in
this) if (predicate(element)) return element\n    return null\n}\n\n**\n * Returns the first element matching the given
[predicate], or `null` if element was not found.\n *^/npublic inline fun ByteArray.firstOrNull(predicate: (Byte) ->
Boolean): Byte? {\n    for (element in this) if (predicate(element)) return element\n    return null\n}\n\n**\n *
Returns the first element matching the given [predicate], or `null` if element was not found.\n *^/npublic inline fun
ShortArray.firstOrNull(predicate: (Short) -> Boolean): Short? {\n    for (element in this) if (predicate(element))
return element\n    return null\n}\n\n**\n * Returns the first element matching the given [predicate], or `null` if
element was not found.\n *^/npublic inline fun IntArray.firstOrNull(predicate: (Int) -> Boolean): Int? {\n    for
(element in this) if (predicate(element)) return element\n    return null\n}\n\n**\n * Returns the first element
matching the given [predicate], or `null` if element was not found.\n *^/npublic inline fun
LongArray.firstOrNull(predicate: (Long) -> Boolean): Long? {\n    for (element in this) if (predicate(element))
return element\n    return null\n}\n\n**\n * Returns the first element matching the given [predicate], or `null` if
element was not found.\n *^/npublic inline fun FloatArray.firstOrNull(predicate: (Float) -> Boolean): Float? {\n
for (element in this) if (predicate(element)) return element\n    return null\n}\n\n**\n * Returns the first element
matching the given [predicate], or `null` if element was not found.\n *^/npublic inline fun
DoubleArray.firstOrNull(predicate: (Double) -> Boolean): Double? {\n    for (element in this) if
(predicate(element)) return element\n    return null\n}\n\n**\n * Returns the first element matching the given
[predicate], or `null` if element was not found.\n *^/npublic inline fun BooleanArray.firstOrNull(predicate:
(Boolean) -> Boolean): Boolean? {\n    for (element in this) if (predicate(element)) return element\n    return
null\n}\n\n**\n * Returns the first element matching the given [predicate], or `null` if element was not found.\n
*^/npublic inline fun CharArray.firstOrNull(predicate: (Char) -> Boolean): Char? {\n    for (element in this) if
(predicate(element)) return element\n    return null\n}\n\n**\n * Returns an element at the given [index] or the
result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n
*^/n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out T>.getOrNull(index: Int, defaultValue: (Int) ->
T): T? {\n    return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n**\n * Returns
an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of
this array.\n *^/n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.getOrNull(index: Int, defaultValue: (Int) -
> Byte): Byte? {\n    return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n**\n *
Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of
bounds of this array.\n *^/n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.getOrNull(index: Int,
defaultValue: (Int) -> Short): Short? {\n    return if (index >= 0 && index <= lastIndex) get(index) else
defaultValue(index)\n}\n\n**\n * Returns an element at the given [index] or the result of calling the [defaultValue]
function if the [index] is out of bounds of this array.\n *^/n@kotlin.internal.InlineOnly\npublic inline fun

```

```

IntArray.getOrElse(index: Int, defaultValue: (Int) -> Int): Int {\n  return if (index >= 0 && index <= lastIndex)
get(index) else defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or the result of calling the
[defaultValue] function if the [index] is out of bounds of this array.\n */\n@kotlin.internal.InlineOnly\npublic inline
fun LongArray.getOrElse(index: Int, defaultValue: (Int) -> Long): Long {\n  return if (index >= 0 && index <=
lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or the result of
calling the [defaultValue] function if the [index] is out of bounds of this array.\n */\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.getOrElse(index: Int, defaultValue: (Int) -> Float):
Float {\n  return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns
an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of
this array.\n */\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.getOrElse(index: Int, defaultValue:
(Int) -> Double): Double {\n  return if (index >= 0 && index <= lastIndex) get(index) else
defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or the result of calling the [defaultValue]
function if the [index] is out of bounds of this array.\n */\n@kotlin.internal.InlineOnly\npublic inline fun
BooleanArray.getOrElse(index: Int, defaultValue: (Int) -> Boolean): Boolean {\n  return if (index >= 0 && index <=
lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or the result
of calling the [defaultValue] function if the [index] is out of bounds of this array.\n */\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.getOrElse(index: Int, defaultValue: (Int) -> Char):
Char {\n  return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns
an element at the given [index] or `null` if the [index] is out of bounds of this array.\n */\n * @sample
samples.collections.Collections.Elements.getOrNull\n */\npublic fun <T> Array<out T>.getOrNull(index: Int): T?
{\n  return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns an element at the
given [index] or `null` if the [index] is out of bounds of this array.\n */\n * @sample
samples.collections.Collections.Elements.getOrNull\n */\npublic fun ByteArray.getOrNull(index: Int): Byte? {\n
return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns an element at the given
[index] or `null` if the [index] is out of bounds of this array.\n */\n * @sample
samples.collections.Collections.Elements.getOrNull\n */\npublic fun ShortArray.getOrNull(index: Int): Short? {\n
return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns an element at the given
[index] or `null` if the [index] is out of bounds of this array.\n */\n * @sample
samples.collections.Collections.Elements.getOrNull\n */\npublic fun IntArray.getOrNull(index: Int): Int? {\n
return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns an element at the given
[index] or `null` if the [index] is out of bounds of this array.\n */\n * @sample
samples.collections.Collections.Elements.getOrNull\n */\npublic fun LongArray.getOrNull(index: Int): Long? {\n
return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns an element at the given
[index] or `null` if the [index] is out of bounds of this array.\n */\n * @sample
samples.collections.Collections.Elements.getOrNull\n */\npublic fun FloatArray.getOrNull(index: Int): Float? {\n
return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns an element at the given
[index] or `null` if the [index] is out of bounds of this array.\n */\n * @sample
samples.collections.Collections.Elements.getOrNull\n */\npublic fun DoubleArray.getOrNull(index: Int): Double?
{\n  return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns an element at the
given [index] or `null` if the [index] is out of bounds of this array.\n */\n * @sample
samples.collections.Collections.Elements.getOrNull\n */\npublic fun BooleanArray.getOrNull(index: Int): Boolean?
{\n  return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns an element at the
given [index] or `null` if the [index] is out of bounds of this array.\n */\n * @sample
samples.collections.Collections.Elements.getOrNull\n */\npublic fun CharArray.getOrNull(index: Int): Char? {\n
return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns first index of [element], or -
1 if the array does not contain element.\n */\npublic fun <@kotlin.internal.OnlyInputTypes T> Array<out
T>.indexOf(element: T): Int {\n  if (element == null) {\n    for (index in indices) {\n      if (this[index] ==
null) {\n        return index\n      }\n    }\n  } else {\n    for (index in indices) {\n      if (element ==

```

```

this[index]) {\n        return index\n    }\n }\n }\n return -1\n}\n\n/**\n * Returns first index of
[element], or -1 if the array does not contain element.\n */\npublic fun ByteArray.indexOf(element: Byte): Int {\n
for (index in indices) {\n    if (element == this[index]) {\n        return index\n    }\n }\n return -
1\n}\n\n/**\n * Returns first index of [element], or -1 if the array does not contain element.\n */\npublic fun
ShortArray.indexOf(element: Short): Int {\n    for (index in indices) {\n        if (element == this[index]) {\n
return index\n    }\n }\n return -1\n}\n\n/**\n * Returns first index of [element], or -1 if the array does not
contain element.\n */\npublic fun IntArray.indexOf(element: Int): Int {\n    for (index in indices) {\n        if (element
== this[index]) {\n            return index\n        }\n    }\n return -1\n}\n\n/**\n * Returns first index of [element], or -
1 if the array does not contain element.\n */\npublic fun LongArray.indexOf(element: Long): Int {\n    for (index in
indices) {\n        if (element == this[index]) {\n            return index\n        }\n    }\n return -1\n}\n\n/**\n * Returns
first index of [element], or -1 if the array does not contain element.\n */\n@Deprecated("The function has unclear
behavior when searching for NaN or zero values and will be removed soon. Use 'indexOfFirst { it == element }'
instead to continue using this behavior, or '.asList().indexOf(element: T)' to get the same search behavior as in a
list.", ReplaceWith("indexOfFirst { it == element }"))\n@DeprecatedSinceKotlin(warningSince = "1.4",
errorSince = "1.6")\npublic fun FloatArray.indexOf(element: Float): Int {\n    for (index in indices) {\n        if
(element == this[index]) {\n            return index\n        }\n    }\n return -1\n}\n\n/**\n * Returns first index of
[element], or -1 if the array does not contain element.\n */\n@Deprecated("The function has unclear behavior when
searching for NaN or zero values and will be removed soon. Use 'indexOfFirst { it == element }' instead to continue
using this behavior, or '.asList().indexOf(element: T)' to get the same search behavior as in a list.",
ReplaceWith("indexOfFirst { it == element }"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.6")\npublic fun DoubleArray.indexOf(element: Double): Int {\n    for (index in indices) {\n        if (element ==
this[index]) {\n            return index\n        }\n    }\n return -1\n}\n\n/**\n * Returns first index of [element], or -1 if
the array does not contain element.\n */\npublic fun BooleanArray.indexOf(element: Boolean): Int {\n    for (index
in indices) {\n        if (element == this[index]) {\n            return index\n        }\n    }\n return -1\n}\n\n/**\n *
Returns first index of [element], or -1 if the array does not contain element.\n */\npublic fun
CharArray.indexOf(element: Char): Int {\n    for (index in indices) {\n        if (element == this[index]) {\n
return index\n    }\n }\n return -1\n}\n\n/**\n * Returns index of the first element matching the given
[predicate], or -1 if the array does not contain such element.\n */\npublic inline fun <T> Array<out
T>.indexOfFirst(predicate: (T) -> Boolean): Int {\n    for (index in indices) {\n        if (predicate(this[index])) {\n
return index\n    }\n }\n return -1\n}\n\n/**\n * Returns index of the first element matching the given
[predicate], or -1 if the array does not contain such element.\n */\npublic inline fun
ByteArray.indexOfFirst(predicate: (Byte) -> Boolean): Int {\n    for (index in indices) {\n        if
(predicate(this[index])) {\n            return index\n        }\n    }\n return -1\n}\n\n/**\n * Returns index of the first
element matching the given [predicate], or -1 if the array does not contain such element.\n */\npublic inline fun
ShortArray.indexOfFirst(predicate: (Short) -> Boolean): Int {\n    for (index in indices) {\n        if
(predicate(this[index])) {\n            return index\n        }\n    }\n return -1\n}\n\n/**\n * Returns index of the first
element matching the given [predicate], or -1 if the array does not contain such element.\n */\npublic inline fun
IntArray.indexOfFirst(predicate: (Int) -> Boolean): Int {\n    for (index in indices) {\n        if (predicate(this[index]))
{\n            return index\n        }\n    }\n return -1\n}\n\n/**\n * Returns index of the first element matching the
given [predicate], or -1 if the array does not contain such element.\n */\npublic inline fun
LongArray.indexOfFirst(predicate: (Long) -> Boolean): Int {\n    for (index in indices) {\n        if
(predicate(this[index])) {\n            return index\n        }\n    }\n return -1\n}\n\n/**\n * Returns index of the first
element matching the given [predicate], or -1 if the array does not contain such element.\n */\npublic inline fun
FloatArray.indexOfFirst(predicate: (Float) -> Boolean): Int {\n    for (index in indices) {\n        if
(predicate(this[index])) {\n            return index\n        }\n    }\n return -1\n}\n\n/**\n * Returns index of the first
element matching the given [predicate], or -1 if the array does not contain such element.\n */\npublic inline fun
DoubleArray.indexOfFirst(predicate: (Double) -> Boolean): Int {\n    for (index in indices) {\n        if
(predicate(this[index])) {\n            return index\n        }\n    }\n return -1\n}\n\n/**\n * Returns index of the first

```

```

element matching the given [predicate], or -1 if the array does not contain such element.\n *\npublic inline fun
BooleanArray.indexOfFirst(predicate: (Boolean) -> Boolean): Int {\n for (index in indices) {\n if
(predicate(this[index])) {\n return index\n }\n }\n return -1\n}\n\n/**\n * Returns index of the first
element matching the given [predicate], or -1 if the array does not contain such element.\n *\npublic inline fun
CharArray.indexOfFirst(predicate: (Char) -> Boolean): Int {\n for (index in indices) {\n if
(predicate(this[index])) {\n return index\n }\n }\n return -1\n}\n\n/**\n * Returns index of the last
element matching the given [predicate], or -1 if the array does not contain such element.\n *\npublic inline fun <T>
Array<out T>.indexOfLast(predicate: (T) -> Boolean): Int {\n for (index in indices.reversed()) {\n if
(predicate(this[index])) {\n return index\n }\n }\n return -1\n}\n\n/**\n * Returns index of the last
element matching the given [predicate], or -1 if the array does not contain such element.\n *\npublic inline fun
ByteArray.indexOfLast(predicate: (Byte) -> Boolean): Int {\n for (index in indices.reversed()) {\n if
(predicate(this[index])) {\n return index\n }\n }\n return -1\n}\n\n/**\n * Returns index of the last
element matching the given [predicate], or -1 if the array does not contain such element.\n *\npublic inline fun
ShortArray.indexOfLast(predicate: (Short) -> Boolean): Int {\n for (index in indices.reversed()) {\n if
(predicate(this[index])) {\n return index\n }\n }\n return -1\n}\n\n/**\n * Returns index of the last
element matching the given [predicate], or -1 if the array does not contain such element.\n *\npublic inline fun
IntArray.indexOfLast(predicate: (Int) -> Boolean): Int {\n for (index in indices.reversed()) {\n if
(predicate(this[index])) {\n return index\n }\n }\n return -1\n}\n\n/**\n * Returns index of the last
element matching the given [predicate], or -1 if the array does not contain such element.\n *\npublic inline fun
LongArray.indexOfLast(predicate: (Long) -> Boolean): Int {\n for (index in indices.reversed()) {\n if
(predicate(this[index])) {\n return index\n }\n }\n return -1\n}\n\n/**\n * Returns index of the last
element matching the given [predicate], or -1 if the array does not contain such element.\n *\npublic inline fun
FloatArray.indexOfLast(predicate: (Float) -> Boolean): Int {\n for (index in indices.reversed()) {\n if
(predicate(this[index])) {\n return index\n }\n }\n return -1\n}\n\n/**\n * Returns index of the last
element matching the given [predicate], or -1 if the array does not contain such element.\n *\npublic inline fun
DoubleArray.indexOfLast(predicate: (Double) -> Boolean): Int {\n for (index in indices.reversed()) {\n if
(predicate(this[index])) {\n return index\n }\n }\n return -1\n}\n\n/**\n * Returns index of the last
element matching the given [predicate], or -1 if the array does not contain such element.\n *\npublic inline fun
BooleanArray.indexOfLast(predicate: (Boolean) -> Boolean): Int {\n for (index in indices.reversed()) {\n if
(predicate(this[index])) {\n return index\n }\n }\n return -1\n}\n\n/**\n * Returns index of the last
element matching the given [predicate], or -1 if the array does not contain such element.\n *\npublic inline fun
CharArray.indexOfLast(predicate: (Char) -> Boolean): Int {\n for (index in indices.reversed()) {\n if
(predicate(this[index])) {\n return index\n }\n }\n return -1\n}\n\n/**\n * Returns the last element.\n *
\n * @throws NoSuchElementException if the array is empty.\n * \n * @sample
\n * \n * @throws NoSuchElementException if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n *\npublic fun <T> Array<out T>.last(): T {\n if (isEmpty())\n
throw NoSuchElementException("Array is empty.")\n return this[lastIndex]\n}\n\n/**\n * Returns the last
element.\n * \n * @throws NoSuchElementException if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n *\npublic fun ByteArray.last(): Byte {\n if (isEmpty())\n
throw NoSuchElementException("Array is empty.")\n return this[lastIndex]\n}\n\n/**\n * Returns the last
element.\n * \n * @throws NoSuchElementException if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n *\npublic fun ShortArray.last(): Short {\n if (isEmpty())\n
throw NoSuchElementException("Array is empty.")\n return this[lastIndex]\n}\n\n/**\n * Returns the last
element.\n * \n * @throws NoSuchElementException if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n *\npublic fun IntArray.last(): Int {\n if (isEmpty())\n
throw NoSuchElementException("Array is empty.")\n return this[lastIndex]\n}\n\n/**\n * Returns the last
element.\n * \n * @throws NoSuchElementException if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n *\npublic fun LongArray.last(): Long {\n if (isEmpty())\n
throw NoSuchElementException("Array is empty.")\n return this[lastIndex]\n}\n\n/**\n * Returns the last

```

```

element.\n * \n * @throws NoSuchElementException if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic fun FloatArray.last(): Float {\n  if (isEmpty())\n  throw NoSuchElementException("Array is empty.")\n  return this[lastIndex]\n}\n\n/**\n * Returns the last
element.\n * \n * @throws NoSuchElementException if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic fun DoubleArray.last(): Double {\n  if (isEmpty())\n  throw NoSuchElementException("Array is empty.")\n  return this[lastIndex]\n}\n\n/**\n * Returns the last
element.\n * \n * @throws NoSuchElementException if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic fun BooleanArray.last(): Boolean {\n  if (isEmpty())\n  throw NoSuchElementException("Array is empty.")\n  return this[lastIndex]\n}\n\n/**\n * Returns the last
element.\n * \n * @throws NoSuchElementException if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic fun CharArray.last(): Char {\n  if (isEmpty())\n  throw NoSuchElementException("Array is empty.")\n  return this[lastIndex]\n}\n\n/**\n * Returns the last
element matching the given [predicate].\n * \n * @throws NoSuchElementException if no such element is found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic inline fun <T> Array<out
T>.last(predicate: (T) -> Boolean): T {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if (predicate(element)) return element\n  }\n  throw NoSuchElementException("Array contains no element
matching the predicate.")\n}\n\n/**\n * Returns the last element matching the given [predicate].\n * \n * @throws
NoSuchElementException if no such element is found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic inline fun ByteArray.last(predicate: (Byte) -> Boolean):
Byte {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if (predicate(element)) return
element\n  }\n  throw NoSuchElementException("Array contains no element matching the
predicate.")\n}\n\n/**\n * Returns the last element matching the given [predicate].\n * \n * @throws
NoSuchElementException if no such element is found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic inline fun ShortArray.last(predicate: (Short) -> Boolean):
Short {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if (predicate(element))
return element\n  }\n  throw NoSuchElementException("Array contains no element matching the
predicate.")\n}\n\n/**\n * Returns the last element matching the given [predicate].\n * \n * @throws
NoSuchElementException if no such element is found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic inline fun IntArray.last(predicate: (Int) -> Boolean): Int
{\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if (predicate(element)) return
element\n  }\n  throw NoSuchElementException("Array contains no element matching the
predicate.")\n}\n\n/**\n * Returns the last element matching the given [predicate].\n * \n * @throws
NoSuchElementException if no such element is found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic inline fun LongArray.last(predicate: (Long) -> Boolean):
Long {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if (predicate(element))
return element\n  }\n  throw NoSuchElementException("Array contains no element matching the
predicate.")\n}\n\n/**\n * Returns the last element matching the given [predicate].\n * \n * @throws
NoSuchElementException if no such element is found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic inline fun FloatArray.last(predicate: (Float) -> Boolean):
Float {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if (predicate(element))
return element\n  }\n  throw NoSuchElementException("Array contains no element matching the
predicate.")\n}\n\n/**\n * Returns the last element matching the given [predicate].\n * \n * @throws
NoSuchElementException if no such element is found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic inline fun DoubleArray.last(predicate: (Double) ->
Boolean): Double {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if
(predicate(element)) return element\n  }\n  throw NoSuchElementException("Array contains no element
matching the predicate.")\n}\n\n/**\n * Returns the last element matching the given [predicate].\n * \n * @throws
NoSuchElementException if no such element is found.\n * \n * @sample

```

```

samples.collections.Collections.Elements.last\n *^/npublic inline fun BooleanArray.last(predicate: (Boolean) ->
Boolean): Boolean {\n for (index in this.indices.reversed()) {\n val element = this[index]\n if
(predicate(element)) return element\n }\n throw NoSuchElementException("Array contains no element
matching the predicate.")\n}\n/n/n/**\n * Returns the last element matching the given [predicate].\n * \n * @throws
NoSuchElementException if no such element is found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic inline fun CharArray.last(predicate: (Char) -> Boolean):
Char {\n for (index in this.indices.reversed()) {\n val element = this[index]\n if (predicate(element))
return element\n }\n throw NoSuchElementException("Array contains no element matching the
predicate.")\n}\n/n/n/**\n * Returns last index of [element], or -1 if the array does not contain element.\n *^/npublic
fun <kotlin.internal.OnlyInputTypes T> Array<out T>.lastIndexOf(element: T): Int {\n if (element == null) {\n
for (index in indices.reversed()) {\n if (this[index] == null) {\n return index\n }\n }\n } else {\n for (index in indices.reversed()) {\n if (element == this[index]) {\n return index\n }\n }\n }\n return -1\n}\n/n/n/**\n * Returns last index of [element], or -1 if the array does not contain
element.\n *^/npublic fun ByteArray.lastIndexOf(element: Byte): Int {\n for (index in indices.reversed()) {\n if
(element == this[index]) {\n return index\n }\n }\n return -1\n}\n/n/n/**\n * Returns last index of
[element], or -1 if the array does not contain element.\n *^/npublic fun ShortArray.lastIndexOf(element: Short): Int
{\n for (index in indices.reversed()) {\n if (element == this[index]) {\n return index\n }\n }\n
return -1\n}\n/n/n/**\n * Returns last index of [element], or -1 if the array does not contain element.\n *^/npublic fun
IntArray.lastIndexOf(element: Int): Int {\n for (index in indices.reversed()) {\n if (element == this[index]) {\n
return index\n }\n }\n return -1\n}\n/n/n/**\n * Returns last index of [element], or -1 if the array does not
contain element.\n *^/npublic fun LongArray.lastIndexOf(element: Long): Int {\n for (index in indices.reversed())
{\n if (element == this[index]) {\n return index\n }\n }\n return -1\n}\n/n/n/**\n * Returns last
index of [element], or -1 if the array does not contain element.\n *^/n@Deprecated("The function has unclear
behavior when searching for NaN or zero values and will be removed soon. Use 'indexOfLast { it == element }'
instead to continue using this behavior, or '.asList().lastIndexOf(element: T)' to get the same search behavior as in a
list.", ReplaceWith("indexOfLast { it == element }"))\n@DeprecatedSinceKotlin(warningSince = "1.4",
errorSince = "1.6")\npublic fun FloatArray.lastIndexOf(element: Float): Int {\n for (index in indices.reversed())
{\n if (element == this[index]) {\n return index\n }\n }\n return -1\n}\n/n/n/**\n * Returns last
index of [element], or -1 if the array does not contain element.\n *^/n@Deprecated("The function has unclear
behavior when searching for NaN or zero values and will be removed soon. Use 'indexOfLast { it == element }'
instead to continue using this behavior, or '.asList().lastIndexOf(element: T)' to get the same search behavior as in a
list.", ReplaceWith("indexOfLast { it == element }"))\n@DeprecatedSinceKotlin(warningSince = "1.4",
errorSince = "1.6")\npublic fun DoubleArray.lastIndexOf(element: Double): Int {\n for (index in
indices.reversed()) {\n if (element == this[index]) {\n return index\n }\n }\n return -1\n}\n/n/n/**\n *
Returns last index of [element], or -1 if the array does not contain element.\n *^/npublic fun
BooleanArray.lastIndexOf(element: Boolean): Int {\n for (index in indices.reversed()) {\n if (element ==
this[index]) {\n return index\n }\n }\n return -1\n}\n/n/n/**\n * Returns last index of [element], or -1 if
the array does not contain element.\n *^/npublic fun CharArray.lastIndexOf(element: Char): Int {\n for (index in
indices.reversed()) {\n if (element == this[index]) {\n return index\n }\n }\n return -1\n}\n/n/n/**\n *
Returns the last element, or `null` if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^/npublic fun <T> Array<out T>.lastOrNull(): T? {\n return if
(isEmpty()) null else this[size - 1]\n}\n/n/n/**\n * Returns the last element, or `null` if the array is empty.\n * \n *
@sample samples.collections.Collections.Elements.last\n *^/npublic fun ByteArray.lastOrNull(): Byte? {\n return
if (isEmpty()) null else this[size - 1]\n}\n/n/n/**\n * Returns the last element, or `null` if the array is empty.\n * \n *
@sample samples.collections.Collections.Elements.last\n *^/npublic fun ShortArray.lastOrNull(): Short? {\n
return if (isEmpty()) null else this[size - 1]\n}\n/n/n/**\n * Returns the last element, or `null` if the array is empty.\n * \n *
@sample samples.collections.Collections.Elements.last\n *^/npublic fun IntArray.lastOrNull(): Int? {\n return
if (isEmpty()) null else this[size - 1]\n}\n/n/n/**\n * Returns the last element, or `null` if the array is empty.\n * \n *

```

```

@sample samples.collections.Collections.Elements.last\n *^\npublic fun LongArray.lastOrNull(): Long? {\n  return
if (isEmpty()) null else this[size - 1]\n}\n\n/**\n * Returns the last element, or `null` if the array is empty.\n * \n *
@sample samples.collections.Collections.Elements.last\n *^\npublic fun FloatArray.lastOrNull(): Float? {\n  return
if (isEmpty()) null else this[size - 1]\n}\n\n/**\n * Returns the last element, or `null` if the array is empty.\n * \n *
@sample samples.collections.Collections.Elements.last\n *^\npublic fun DoubleArray.lastOrNull(): Double? {\n
return if (isEmpty()) null else this[size - 1]\n}\n\n/**\n * Returns the last element, or `null` if the array is empty.\n *
\n * @sample samples.collections.Collections.Elements.last\n *^\npublic fun BooleanArray.lastOrNull(): Boolean?
{\n  return if (isEmpty()) null else this[size - 1]\n}\n\n/**\n * Returns the last element, or `null` if the array is
empty.\n * \n * @sample samples.collections.Collections.Elements.last\n *^\npublic fun CharArray.lastOrNull():
Char? {\n  return if (isEmpty()) null else this[size - 1]\n}\n\n/**\n * Returns the last element matching the given
[predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^\npublic inline fun <T> Array<out T>.lastOrNull(predicate: (T) ->
Boolean): T? {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if
(predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns the last element matching the given
[predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^\npublic inline fun ByteArray.lastOrNull(predicate: (Byte) ->
Boolean): Byte? {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if
(predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns the last element matching the given
[predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^\npublic inline fun ShortArray.lastOrNull(predicate: (Short) ->
Boolean): Short? {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if
(predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns the last element matching the given
[predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^\npublic inline fun IntArray.lastOrNull(predicate: (Int) ->
Boolean): Int? {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if
(predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns the last element matching the given
[predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^\npublic inline fun LongArray.lastOrNull(predicate: (Long) ->
Boolean): Long? {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if
(predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns the last element matching the given
[predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^\npublic inline fun FloatArray.lastOrNull(predicate: (Float) ->
Boolean): Float? {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if
(predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns the last element matching the given
[predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^\npublic inline fun DoubleArray.lastOrNull(predicate: (Double) ->
Boolean): Double? {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if
(predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns the last element matching the given
[predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^\npublic inline fun BooleanArray.lastOrNull(predicate: (Boolean)
-> Boolean): Boolean? {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if
(predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns the last element matching the given
[predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n *^\npublic inline fun CharArray.lastOrNull(predicate: (Char) ->
Boolean): Char? {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if
(predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns a random element from this array.\n
\n * \n * @throws NoSuchElementException if this array is empty.\n
\n * \n * @since Kotlin("1.3")\n\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out T>.random(): T {\n

```

```

return random(Random)\n}\n\n/**\n * Returns a random element from this array.\n * \n * @throws
NoSuchElementException if this array is empty.\n */\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic
inline fun ByteArray.random(): Byte {\n    return random(Random)\n}\n\n/**\n * Returns a random element from
this array.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.random(): Short {\n    return
random(Random)\n}\n\n/**\n * Returns a random element from this array.\n * \n * @throws
NoSuchElementException if this array is empty.\n */\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic
inline fun IntArray.random(): Int {\n    return random(Random)\n}\n\n/**\n * Returns a random element from this
array.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.random(): Long {\n    return
random(Random)\n}\n\n/**\n * Returns a random element from this array.\n * \n * @throws
NoSuchElementException if this array is empty.\n */\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic
inline fun FloatArray.random(): Float {\n    return random(Random)\n}\n\n/**\n * Returns a random element from
this array.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.random(): Double {\n    return
random(Random)\n}\n\n/**\n * Returns a random element from this array.\n * \n * @throws
NoSuchElementException if this array is empty.\n */\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic
inline fun BooleanArray.random(): Boolean {\n    return random(Random)\n}\n\n/**\n * Returns a random element
from this array.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.random(): Char {\n    return
random(Random)\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic fun <T> Array<out T>.random(random: Random): T {\n    if
(isEmpty())\n        throw
NoSuchElementException("Array is empty.")\n    return get(random.nextInt(size))\n}\n\n/**\n * Returns a random
element from this array using the specified source of randomness.\n * \n * @throws NoSuchElementException if
this array is empty.\n */\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic fun ByteArray.random(random:
Random): Byte {\n    if (isEmpty())\n        throw NoSuchElementException("Array is empty.")\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic fun ShortArray.random(random: Random): Short {\n    if
(isEmpty())\n        throw
NoSuchElementException("Array is empty.")\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic fun IntArray.random(random: Random): Int {\n    if
(isEmpty())\n        throw NoSuchElementException("Array is empty.")\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic fun LongArray.random(random: Random): Long {\n    if
(isEmpty())\n        throw
NoSuchElementException("Array is empty.")\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic fun FloatArray.random(random: Random): Float {\n    if
(isEmpty())\n        throw NoSuchElementException("Array is empty.")\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic fun DoubleArray.random(random: Random): Double {\n    if
(isEmpty())\n        throw
NoSuchElementException("Array is empty.")\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic fun BooleanArray.random(random: Random): Boolean {\n    if
(isEmpty())\n        throw NoSuchElementException("Array is empty.")\n    return

```

```

get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness.\n * \n * @throws NoSuchElementException if this array is empty.\n */\n@SinceKotlin("1.3")\npublic
fun CharArray.random(random: Random): Char {\n    if (isEmpty())\n        throw NoSuchElementException("Array
is empty.")\n    return get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array, or `null` if
this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npubli
c inline fun <T> Array<out T>.randomOrNull(): T? {\n    return randomOrNull(Random)\n}\n\n/**\n * Returns a
random element from this array, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npubli
c inline fun ByteArray.randomOrNull(): Byte? {\n    return randomOrNull(Random)\n}\n\n/**\n * Returns a random
element from this array, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npubli
c inline fun ShortArray.randomOrNull(): Short? {\n    return randomOrNull(Random)\n}\n\n/**\n * Returns a
random element from this array, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npubli
c inline fun IntArray.randomOrNull(): Int? {\n    return randomOrNull(Random)\n}\n\n/**\n * Returns a random
element from this array, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npubli
c inline fun LongArray.randomOrNull(): Long? {\n    return randomOrNull(Random)\n}\n\n/**\n * Returns a
random element from this array, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npubli
c inline fun FloatArray.randomOrNull(): Float? {\n    return randomOrNull(Random)\n}\n\n/**\n * Returns a
random element from this array, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npubli
c inline fun DoubleArray.randomOrNull(): Double? {\n    return randomOrNull(Random)\n}\n\n/**\n * Returns a
random element from this array, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npubli
c inline fun BooleanArray.randomOrNull(): Boolean? {\n    return randomOrNull(Random)\n}\n\n/**\n * Returns a
random element from this array, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npubli
c inline fun CharArray.randomOrNull(): Char? {\n    return randomOrNull(Random)\n}\n\n/**\n * Returns a
random element from this array using the specified source of randomness, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun <T> Array<out
T>.randomOrNull(random: Random): T? {\n    if (isEmpty())\n        return null\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
ByteArray.randomOrNull(random: Random): Byte? {\n    if (isEmpty())\n        return null\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
ShortArray.randomOrNull(random: Random): Short? {\n    if (isEmpty())\n        return null\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
IntArray.randomOrNull(random: Random): Int? {\n    if (isEmpty())\n        return null\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness, or `null` if this array is empty.\n

```

```

*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
LongArray.randomOrNull(random: Random): Long? {\n if (isEmpty())\n return null\n return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
FloatArray.randomOrNull(random: Random): Float? {\n if (isEmpty())\n return null\n return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
DoubleArray.randomOrNull(random: Random): Double? {\n if (isEmpty())\n return null\n return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
BooleanArray.randomOrNull(random: Random): Boolean? {\n if (isEmpty())\n return null\n return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
CharArray.randomOrNull(random: Random): Char? {\n if (isEmpty())\n return null\n return
get(random.nextInt(size))\n}\n\n/**\n * Returns the single element, or throws an exception if the array is empty or
has more than one element.\n */\npublic fun <T> Array<out T>.single(): T {\n return when (size) {\n 0 ->
throw NoSuchElementException("Array is empty.")\n 1 -> this[0]\n else -> throw
IllegalArgumentException("Array has more than one element.")\n }\n}\n\n/**\n * Returns the single element, or
throws an exception if the array is empty or has more than one element.\n */\npublic fun ByteArray.single(): Byte
{\n return when (size) {\n 0 -> throw NoSuchElementException("Array is empty.")\n 1 -> this[0]\n
else -> throw IllegalArgumentException("Array has more than one element.")\n }\n}\n\n/**\n * Returns the
single element, or throws an exception if the array is empty or has more than one
element.\n */\npublic fun ShortArray.single(): Short {\n return when (size) {\n 0 -> throw
NoSuchElementException("Array is empty.")\n 1 -> this[0]\n else -> throw
IllegalArgumentException("Array has more than one element.")\n }\n}\n\n/**\n * Returns the single element, or
throws an exception if the array is empty or has more than one
element.\n */\npublic fun IntArray.single(): Int {\n return when (size) {\n 0 -> throw
NoSuchElementException("Array is empty.")\n 1 -> this[0]\n else -> throw
IllegalArgumentException("Array has more than one element.")\n }\n}\n\n/**\n * Returns the single element, or
throws an exception if the array is empty or has more than one element.\n */\npublic fun LongArray.single(): Long
{\n return when (size) {\n 0 -> throw NoSuchElementException("Array is empty.")\n 1 -> this[0]\n
else -> throw IllegalArgumentException("Array has more than one element.")\n }\n}\n\n/**\n * Returns the
single element, or throws an exception if the array is empty or has more than one
element.\n */\npublic fun FloatArray.single(): Float {\n return when (size) {\n 0 -> throw
NoSuchElementException("Array is empty.")\n 1 -> this[0]\n else -> throw
IllegalArgumentException("Array has more than one element.")\n }\n}\n\n/**\n * Returns the single element, or
throws an exception if the array is empty or has more than one
element.\n */\npublic fun DoubleArray.single(): Double {\n return when (size) {\n 0 -> throw
NoSuchElementException("Array is empty.")\n 1 -> this[0]\n else -> throw
IllegalArgumentException("Array has more than one element.")\n }\n}\n\n/**\n * Returns the single element, or
throws an exception if the array is empty or has more than one element.\n */\npublic fun BooleanArray.single():
Boolean {\n return when (size) {\n 0 -> throw NoSuchElementException("Array is empty.")\n 1 ->
this[0]\n else -> throw IllegalArgumentException("Array has more than one element.")\n }\n}\n\n/**\n *
Returns the single element, or throws an exception if the array is empty or has more
than one element.\n */\npublic fun CharArray.single(): Char {\n return when (size) {\n 0 ->
throw NoSuchElementException("Array is empty.")\n 1 -> this[0]\n else -> throw
IllegalArgumentException("Array has more than one element.")\n }\n}

```

```

}
}

Returns the single element matching the given [predicate], or throws exception if there is no or more than one matching element.
public inline fun <T> Array<out T>.single(predicate: (T) -> Boolean): T {
    var single: T? = null
    var found = false
    for (element in this) {
        if (predicate(element)) {
            if (found) throw IllegalArgumentException("Array contains more than one matching element.")
            single = element
            found = true
        }
    }
    if (!found) throw NoSuchElementException("Array contains no element matching the predicate.")
    @Suppress("UNCHECKED_CAST") return single as T
}

Returns the single element matching the given [predicate], or throws exception if there is no or more than one matching element.
public inline fun ByteArray.single(predicate: (Byte) -> Boolean): Byte {
    var single: Byte? = null
    var found = false
    for (element in this) {
        if (predicate(element)) {
            if (found) throw IllegalArgumentException("Array contains more than one matching element.")
            single = element
            found = true
        }
    }
    if (!found) throw NoSuchElementException("Array contains no element matching the predicate.")
    @Suppress("UNCHECKED_CAST") return single as Byte
}

Returns the single element matching the given [predicate], or throws exception if there is no or more than one matching element.
public inline fun ShortArray.single(predicate: (Short) -> Boolean): Short {
    var single: Short? = null
    var found = false
    for (element in this) {
        if (predicate(element)) {
            if (found) throw IllegalArgumentException("Array contains more than one matching element.")
            single = element
            found = true
        }
    }
    if (!found) throw NoSuchElementException("Array contains no element matching the predicate.")
    @Suppress("UNCHECKED_CAST") return single as Short
}

Returns the single element matching the given [predicate], or throws exception if there is no or more than one matching element.
public inline fun IntArray.single(predicate: (Int) -> Boolean): Int {
    var single: Int? = null
    var found = false
    for (element in this) {
        if (predicate(element)) {
            if (found) throw IllegalArgumentException("Array contains more than one matching element.")
            single = element
            found = true
        }
    }
    if (!found) throw NoSuchElementException("Array contains no element matching the predicate.")
    @Suppress("UNCHECKED_CAST") return single as Int
}

Returns the single element matching the given [predicate], or throws exception if there is no or more than one matching element.
public inline fun LongArray.single(predicate: (Long) -> Boolean): Long {
    var single: Long? = null
    var found = false
    for (element in this) {
        if (predicate(element)) {
            if (found) throw IllegalArgumentException("Array contains more than one matching element.")
            single = element
            found = true
        }
    }
    if (!found) throw NoSuchElementException("Array contains no element matching the predicate.")
    @Suppress("UNCHECKED_CAST") return single as Long
}

Returns the single element matching the given [predicate], or throws exception if there is no or more than one matching element.
public inline fun FloatArray.single(predicate: (Float) -> Boolean): Float {
    var single: Float? = null
    var found = false
    for (element in this) {
        if (predicate(element)) {
            if (found) throw IllegalArgumentException("Array contains more than one matching element.")
            single = element
            found = true
        }
    }
    if (!found) throw NoSuchElementException("Array contains no element matching the predicate.")
    @Suppress("UNCHECKED_CAST") return single as Float
}

Returns the single element matching the given [predicate], or throws exception if there is no or more than one matching element.
public inline fun DoubleArray.single(predicate: (Double) -> Boolean): Double {
    var single: Double? = null
    var found = false
    for (element in this) {
        if (predicate(element)) {
            if (found) throw IllegalArgumentException("Array contains more than one matching element.")
            single = element
            found = true
        }
    }
    if (!found) throw NoSuchElementException("Array contains no element matching the predicate.")
    @Suppress("UNCHECKED_CAST") return single as Double
}

Returns the single element matching the given [predicate], or throws exception if there is no or more than one matching element.
public inline fun BooleanArray.single(predicate: (Boolean) -> Boolean): Boolean {
    var single: Boolean? = null
    var found = false
    for (element in this) {
        if (predicate(element)) {
            if (found) throw IllegalArgumentException("Array contains more than one matching element.")
            single = element
            found = true
        }
    }
    if (!found) throw NoSuchElementException("Array contains no element matching the predicate.")
    @Suppress("UNCHECKED_CAST") return single as Boolean
}

```

```

Boolean\n}\n\n/**\n * Returns the single element matching the given [predicate], or throws exception if there is no
or more than one matching element.\n */\npublic inline fun CharArray.single(predicate: (Char) -> Boolean): Char
{\n    var single: Char? = null\n    var found = false\n    for (element in this) {\n        if (predicate(element)) {\n
if (found) throw IllegalArgumentException("Array contains more than one matching element.")\n            single =
element\n                found = true\n                }\n        }\n        if (!found) throw NoSuchElementException("Array contains no
element matching the predicate.")\n    }\n    @Suppress("UNCHECKED_CAST")\n    return single as Char\n}\n\n/**\n * Returns single element, or `null` if the array is empty or has more than one element.\n */\npublic fun <T>
Array<out T>.singleOrNull(): T? {\n    return if (size == 1) this[0] else null\n}\n\n/**\n * Returns single element, or `null` if the array is empty or has more than one element.\n */\npublic fun ByteArray.singleOrNull(): Byte? {\n
return if (size == 1) this[0] else null\n}\n\n/**\n * Returns single element, or `null` if the array is empty or has more
than one element.\n */\npublic fun ShortArray.singleOrNull(): Short? {\n    return if (size == 1) this[0] else
null\n}\n\n/**\n * Returns single element, or `null` if the array is empty or has more than one element.\n */\npublic
fun IntArray.singleOrNull(): Int? {\n    return if (size == 1) this[0] else null\n}\n\n/**\n * Returns single element, or `null` if the array is empty or has more than one element.\n */\npublic fun LongArray.singleOrNull(): Long? {\n
return if (size == 1) this[0] else null\n}\n\n/**\n * Returns single element, or `null` if the array is empty or has more
than one element.\n */\npublic fun FloatArray.singleOrNull(): Float? {\n    return if (size == 1) this[0] else
null\n}\n\n/**\n * Returns single element, or `null` if the array is empty or has more than one element.\n */\npublic
fun DoubleArray.singleOrNull(): Double? {\n    return if (size == 1) this[0] else null\n}\n\n/**\n * Returns single
element, or `null` if the array is empty or has more than one element.\n */\npublic fun BooleanArray.singleOrNull(): Boolean? {\n
return if (size == 1) this[0] else null\n}\n\n/**\n * Returns single element, or `null` if the array is
empty or has more than one element.\n */\npublic fun CharArray.singleOrNull(): Char? {\n    return if (size == 1)
this[0] else null\n}\n\n/**\n * Returns the single element matching the given [predicate], or `null` if element was not
found or more than one element was found.\n */\npublic inline fun <T> Array<out T>.singleOrNull(predicate: (T) -
> Boolean): T? {\n    var single: T? = null\n    var found = false\n    for (element in this) {\n        if
(predicate(element)) {\n            if (found) return null\n                single = element\n                found = true\n                }\n        }\n        if (!found) return null\n    }\n    return single\n}\n\n/**\n * Returns the single element matching the given [predicate], or `null` if element was not found or more than one element was found.\n */\npublic inline fun
ByteArray.singleOrNull(predicate: (Byte) -> Boolean): Byte? {\n    var single: Byte? = null\n    var found = false\n
for (element in this) {\n        if (predicate(element)) {\n            if (found) return null\n                single = element\n                found = true\n                }\n        }\n        if (!found) return null\n    }\n    return single\n}\n\n/**\n * Returns the single element
matching the given [predicate], or `null` if element was not found or more than one element was found.\n */\npublic inline fun ShortArray.singleOrNull(predicate: (Short) -> Boolean): Short? {\n    var single: Short? = null\n    var
found = false\n    for (element in this) {\n        if (predicate(element)) {\n            if (found) return null\n                single = element\n                found = true\n                }\n        }\n        if (!found) return null\n    }\n    return single\n}\n\n/**\n * Returns the
single element matching the given [predicate], or `null` if element was not found or more than one element was
found.\n */\npublic inline fun IntArray.singleOrNull(predicate: (Int) -> Boolean): Int? {\n    var single: Int? = null\n
var found = false\n    for (element in this) {\n        if (predicate(element)) {\n            if (found) return null\n                single = element\n                found = true\n                }\n        }\n        if (!found) return null\n    }\n    return single\n}\n\n/**\n * Returns the single element matching the given [predicate], or `null` if element was not found or more than one element was found.\n */\npublic inline fun LongArray.singleOrNull(predicate: (Long) -> Boolean): Long? {\n    var
single: Long? = null\n    var found = false\n    for (element in this) {\n        if (predicate(element)) {\n            if
(found) return null\n                single = element\n                found = true\n                }\n        }\n        if (!found) return null\n    }\n    return single\n}\n\n/**\n * Returns the single element matching the given [predicate], or `null` if element was not found or
more than one element was found.\n */\npublic inline fun FloatArray.singleOrNull(predicate: (Float) -> Boolean): Float? {\n    var single: Float? = null\n    var found = false\n    for (element in this) {\n        if (predicate(element))
{\n            if (found) return null\n                single = element\n                found = true\n                }\n        }\n        if (!found) return
null\n    }\n    return single\n}\n\n/**\n * Returns the single element matching the given [predicate], or `null` if element
was not found or more than one element was found.\n */\npublic inline fun DoubleArray.singleOrNull(predicate:

```

```

(Double) -> Boolean): Double? {\n  var single: Double? = null\n  var found = false\n  for (element in this) {\n
if (predicate(element)) {\n    if (found) return null\n    single = element\n    found = true\n  }\n
}\n  if (!found) return null\n  return single\n}\n\n/**\n * Returns the single element matching the given
[predicate], or `null` if element was not found or more than one element was found.\n */\npublic inline fun
BooleanArray.singleOrNull(predicate: (Boolean) -> Boolean): Boolean? {\n  var single: Boolean? = null\n  var
found = false\n  for (element in this) {\n    if (predicate(element)) {\n      if (found) return null\n      single
= element\n      found = true\n    }\n  }\n  if (!found) return null\n  return single\n}\n\n/**\n * Returns the
single element matching the given [predicate], or `null` if element was not found or more than one element was
found.\n */\npublic inline fun CharArray.singleOrNull(predicate: (Char) -> Boolean): Char? {\n  var single: Char?
= null\n  var found = false\n  for (element in this) {\n    if (predicate(element)) {\n      if (found) return
null\n      single = element\n      found = true\n    }\n  }\n  if (!found) return null\n  return
single\n}\n\n/**\n * Returns a list containing all elements except first [n] elements.\n */\n * @throws
IllegalArgumentException if [n] is negative.\n */\n * @sample
samples.collections.Collections.Transformations.drop\n */\npublic fun <T> Array<out T>.drop(n: Int): List<T> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except first [n] elements.\n */\n * @throws
IllegalArgumentException if [n] is negative.\n */\n * @sample
samples.collections.Collections.Transformations.drop\n */\npublic fun ByteArray.drop(n: Int): List<Byte> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except first [n] elements.\n */\n * @throws
IllegalArgumentException if [n] is negative.\n */\n * @sample
samples.collections.Collections.Transformations.drop\n */\npublic fun ShortArray.drop(n: Int): List<Short> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except first [n] elements.\n */\n * @throws
IllegalArgumentException if [n] is negative.\n */\n * @sample
samples.collections.Collections.Transformations.drop\n */\npublic fun IntArray.drop(n: Int): List<Int> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except first [n] elements.\n */\n * @throws
IllegalArgumentException if [n] is negative.\n */\n * @sample
samples.collections.Collections.Transformations.drop\n */\npublic fun LongArray.drop(n: Int): List<Long> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except first [n] elements.\n */\n * @throws
IllegalArgumentException if [n] is negative.\n */\n * @sample
samples.collections.Collections.Transformations.drop\n */\npublic fun FloatArray.drop(n: Int): List<Float> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except first [n] elements.\n */\n * @throws
IllegalArgumentException if [n] is negative.\n */\n * @sample
samples.collections.Collections.Transformations.drop\n */\npublic fun DoubleArray.drop(n: Int): List<Double> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except first [n] elements.\n */\n * @throws
IllegalArgumentException if [n] is negative.\n */\n * @sample
samples.collections.Collections.Transformations.drop\n */\npublic fun BooleanArray.drop(n: Int): List<Boolean>
{\n  require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements\n */\n * @throws
IllegalArgumentException if [n] is negative.\n */\n * @sample
samples.collections.Collections.Transformations.drop\n */\npublic fun CharArray.drop(n: Int): List<Char> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except last [n] elements.\n */\n * @throws

```

```

IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic fun <T> Array<out T>.dropLast(n: Int): List<T> {
\n    require(n >= 0) { \"Requested element count $n is less than zero.\" }\n    return take((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic fun ByteArray.dropLast(n: Int): List<Byte> {
\n    require(n >= 0) { \"Requested element count $n is less than zero.\" }\n    return take((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic fun ShortArray.dropLast(n: Int): List<Short> {
\n    require(n >= 0) { \"Requested element count $n is less than zero.\" }\n    return take((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic fun IntArray.dropLast(n: Int): List<Int> {
\n    require(n >= 0) { \"Requested element count $n is less than zero.\" }\n    return take((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic fun LongArray.dropLast(n: Int): List<Long> {
\n    require(n >= 0) { \"Requested element count $n is less than zero.\" }\n    return take((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic fun FloatArray.dropLast(n: Int): List<Float> {
\n    require(n >= 0) { \"Requested element count $n is less than zero.\" }\n    return take((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic fun DoubleArray.dropLast(n: Int): List<Double>
\n    require(n >= 0) { \"Requested element count $n is less than zero.\" }\n    return take((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic fun BooleanArray.dropLast(n: Int):
List<Boolean> {
\n    require(n >= 0) { \"Requested element count $n is less than zero.\" }\n    return take((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic fun CharArray.dropLast(n: Int): List<Char> {
\n    require(n >= 0) { \"Requested element count $n is less than zero.\" }\n    return take((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except last elements that satisfy the given
[predicate].\n * \n * @sample samples.collections.Collections.Transformations.drop\n *\npublic inline fun <T>
Array<out T>.dropLastWhile(predicate: (T) -> Boolean): List<T> {
\n    for (index in lastIndex downTo 0) {\n        if (!predicate(this[index])) {\n            return take(index + 1)\n        }\n    }\n    return emptyList()\n}\n\n/**\n * Returns
a list containing all elements except last elements that satisfy the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun ByteArray.dropLastWhile(predicate:
(Byte) -> Boolean): List<Byte> {
\n    for (index in lastIndex downTo 0) {\n        if (!predicate(this[index])) {\n            return take(index + 1)\n        }\n    }\n    return emptyList()\n}\n\n/**\n * Returns a list containing all elements
except last elements that satisfy the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun ShortArray.dropLastWhile(predicate:
(Short) -> Boolean): List<Short> {
\n    for (index in lastIndex downTo 0) {\n        if (!predicate(this[index])) {\n            return take(index + 1)\n        }\n    }\n    return emptyList()\n}\n\n/**\n * Returns a list containing all elements
except last elements that satisfy the given [predicate].\n * \n * @sample

```

```

samples.collections.Collections.Transformations.drop\n *\npublic inline fun IntArray.dropLastWhile(predicate:
(Int) -> Boolean): List<Int> {\n  for (index in lastIndex downTo 0) {\n    if (!predicate(this[index])) {\n
return take(index + 1)\n    }\n  }\n  return emptyList()\n}\n\n/**\n * Returns a list containing all elements
except last elements that satisfy the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun LongArray.dropLastWhile(predicate:
(Long) -> Boolean): List<Long> {\n  for (index in lastIndex downTo 0) {\n    if (!predicate(this[index])) {\n
return take(index + 1)\n    }\n  }\n  return emptyList()\n}\n\n/**\n * Returns a list containing all elements
except last elements that satisfy the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun FloatArray.dropLastWhile(predicate:
(Float) -> Boolean): List<Float> {\n  for (index in lastIndex downTo 0) {\n    if (!predicate(this[index])) {\n
return take(index + 1)\n    }\n  }\n  return emptyList()\n}\n\n/**\n * Returns a list containing all elements
except last elements that satisfy the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun DoubleArray.dropLastWhile(predicate:
(Double) -> Boolean): List<Double> {\n  for (index in lastIndex downTo 0) {\n    if (!predicate(this[index])) {\n
return take(index + 1)\n    }\n  }\n  return emptyList()\n}\n\n/**\n * Returns a list containing all elements
except last elements that satisfy the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun
BooleanArray.dropLastWhile(predicate: (Boolean) -> Boolean): List<Boolean> {\n  for (index in lastIndex
downTo 0) {\n    if (!predicate(this[index])) {\n      return take(index + 1)\n    }\n  }\n  return
emptyList()\n}\n\n/**\n * Returns a list containing all elements except last elements that satisfy the given
[predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun
CharArray.dropLastWhile(predicate: (Char) -> Boolean): List<Char> {\n  for (index in lastIndex downTo 0) {\n
if (!predicate(this[index])) {\n    return take(index + 1)\n  }\n}\n  return emptyList()\n}\n\n/**\n *
Returns a list containing all elements except first elements that satisfy the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun
<T> Array<out T>.dropWhile(predicate: (T) -> Boolean): List<T> {\n  var yielding = false\n  val list = ArrayList<T>()\n
for (item in this)\n  if (yielding)\n    list.add(item)\n  else if (!predicate(item)) {\n    list.add(item)\n
yielding = true\n  }\n  return list\n}\n\n/**\n * Returns a list containing all elements except first elements that
satisfy the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun
ByteArray.dropWhile(predicate: (Byte) -> Boolean): List<Byte> {\n  var yielding = false\n  val list =
ArrayList<Byte>()\n  for (item in this)\n  if (yielding)\n    list.add(item)\n  else if (!predicate(item)) {\n
list.add(item)\n    yielding = true\n  }\n  return list\n}\n\n/**\n * Returns a list containing all
elements except first elements that satisfy the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun
ShortArray.dropWhile(predicate:
(Short) -> Boolean): List<Short> {\n  var yielding = false\n  val list = ArrayList<Short>()\n  for (item in this)\n
if (yielding)\n    list.add(item)\n  else if (!predicate(item)) {\n    list.add(item)\n    yielding =
true\n  }\n  return list\n}\n\n/**\n * Returns a list containing all elements except first elements that satisfy the
given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun
IntArray.dropWhile(predicate: (Int) -> Boolean): List<Int> {\n  var yielding = false\n  val list =
ArrayList<Int>()\n  for (item in this)\n  if (yielding)\n    list.add(item)\n  else if (!predicate(item)) {\n
list.add(item)\n    yielding = true\n  }\n  return list\n}\n\n/**\n * Returns a list containing all elements
except first elements that satisfy the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun
LongArray.dropWhile(predicate:
(Long) -> Boolean): List<Long> {\n  var yielding = false\n  val list = ArrayList<Long>()\n  for (item in this)\n
if (yielding)\n    list.add(item)\n  else if (!predicate(item)) {\n    list.add(item)\n    yielding =
true\n  }\n  return list\n}\n\n/**\n * Returns a list containing all elements except first elements that satisfy the
given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n *\npublic inline fun
FloatArray.dropWhile(predicate: (Float) -> Boolean): List<Float> {\n  var yielding = false\n  val list =

```

```

ArrayList<Float>()\n for (item in this)\n     if (yielding)\n         list.add(item)\n     else if (!predicate(item))\n         {\n             list.add(item)\n             yielding = true\n         }\n return list\n}\n\n/**\n * Returns a list containing all elements except first elements that satisfy the given [predicate].\n * \n * @sample\n samples.collections.Collections.Transformations.drop\n */\n\npublic inline fun DoubleArray.dropWhile(predicate: (Double) -> Boolean): List<Double> {\n    var yielding = false\n    val list = ArrayList<Double>()\n    for (item in this)\n        if (yielding)\n            list.add(item)\n        else if (!predicate(item)) {\n            list.add(item)\n            yielding = true\n        }\n    return list\n}\n\n/**\n * Returns a list containing all elements except first elements that satisfy the given [predicate].\n * \n * @sample\n samples.collections.Collections.Transformations.drop\n */\n\npublic inline fun BooleanArray.dropWhile(predicate: (Boolean) -> Boolean): List<Boolean> {\n    var yielding = false\n    val list = ArrayList<Boolean>()\n    for (item in this)\n        if (yielding)\n            list.add(item)\n        else if (!predicate(item)) {\n            list.add(item)\n            yielding = true\n        }\n    return list\n}\n\n/**\n * Returns a list containing all elements except first elements that satisfy the given [predicate].\n * \n * @sample\n samples.collections.Collections.Transformations.drop\n */\n\npublic inline fun CharArray.dropWhile(predicate: (Char) -> Boolean): List<Char> {\n    var yielding = false\n    val list = ArrayList<Char>()\n    for (item in this)\n        if (yielding)\n            list.add(item)\n        else if (!predicate(item)) {\n            list.add(item)\n            yielding = true\n        }\n    return list\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * \n * @sample\n samples.collections.Collections.Filtering.filter\n */\n\npublic inline fun <T> Array<out T>.filter(predicate: (T) -> Boolean): List<T> {\n    return filterTo(ArrayList<T>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * \n * @sample\n samples.collections.Collections.Filtering.filter\n */\n\npublic inline fun ByteArray.filter(predicate: (Byte) -> Boolean): List<Byte> {\n    return filterTo(ArrayList<Byte>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * \n * @sample\n samples.collections.Collections.Filtering.filter\n */\n\npublic inline fun ShortArray.filter(predicate: (Short) -> Boolean): List<Short> {\n    return filterTo(ArrayList<Short>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * \n * @sample\n samples.collections.Collections.Filtering.filter\n */\n\npublic inline fun IntArray.filter(predicate: (Int) -> Boolean): List<Int> {\n    return filterTo(ArrayList<Int>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * \n * @sample\n samples.collections.Collections.Filtering.filter\n */\n\npublic inline fun LongArray.filter(predicate: (Long) -> Boolean): List<Long> {\n    return filterTo(ArrayList<Long>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * \n * @sample\n samples.collections.Collections.Filtering.filter\n */\n\npublic inline fun FloatArray.filter(predicate: (Float) -> Boolean): List<Float> {\n    return filterTo(ArrayList<Float>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * \n * @sample\n samples.collections.Collections.Filtering.filter\n */\n\npublic inline fun DoubleArray.filter(predicate: (Double) -> Boolean): List<Double> {\n    return filterTo(ArrayList<Double>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * \n * @sample\n samples.collections.Collections.Filtering.filter\n */\n\npublic inline fun BooleanArray.filter(predicate: (Boolean) -> Boolean): List<Boolean> {\n    return filterTo(ArrayList<Boolean>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * \n * @sample\n samples.collections.Collections.Filtering.filter\n */\n\npublic inline fun CharArray.filter(predicate: (Char) -> Boolean): List<Char> {\n    return filterTo(ArrayList<Char>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * \n * @param [predicate] function that takes the index of an element and the element itself\n * and returns the result of predicate evaluation on the element.\n * \n * @sample\n samples.collections.Collections.Filtering.filterIndexed\n */\n\npublic inline fun <T> Array<out T>.filterIndexed(predicate: (index: Int, T) -> Boolean): List<T> {\n    return filterIndexedTo(ArrayList<T>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * \n * @param [predicate] function that takes the index of an element and the element itself\n * and returns the result of predicate evaluation on the element.\n * \n * @sample\n samples.collections.Collections.Filtering.filterIndexed\n */\n\npublic inline fun ByteArray.filterIndexed(predicate: (index: Int, Byte) -> Boolean): List<Byte> {\n    return filterIndexedTo(ArrayList<Byte>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the

```

```

given [predicate].\n * @param [predicate] function that takes the index of an element and the element itself\n * and
returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexed\n */\npublic inline fun ShortArray.filterIndexed(predicate:
(index: Int, Short) -> Boolean): List<Short> {\n    return filterIndexedTo(ArrayList<Short>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * @param [predicate] function that takes
the index of an element and the element itself\n * and returns the result of predicate evaluation on the element.\n * \n *
@sample samples.collections.Collections.Filtering.filterIndexed\n */\npublic inline fun
IntArray.filterIndexed(predicate: (index: Int, Int) -> Boolean): List<Int> {\n    return
filterIndexedTo(ArrayList<Int>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the
given [predicate].\n * @param [predicate] function that takes the index of an element and the element itself\n * and
returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexed\n */\npublic inline fun LongArray.filterIndexed(predicate:
(index: Int, Long) -> Boolean): List<Long> {\n    return filterIndexedTo(ArrayList<Long>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * @param [predicate] function that takes
the index of an element and the element itself\n * and returns the result of predicate evaluation on the element.\n * \n *
@sample samples.collections.Collections.Filtering.filterIndexed\n */\npublic inline fun
FloatArray.filterIndexed(predicate: (index: Int, Float) -> Boolean): List<Float> {\n    return
filterIndexedTo(ArrayList<Float>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the
given [predicate].\n * @param [predicate] function that takes the index of an element and the element itself\n * and
returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexed\n */\npublic inline fun DoubleArray.filterIndexed(predicate:
(index: Int, Double) -> Boolean): List<Double> {\n    return filterIndexedTo(ArrayList<Double>(),
predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * @param
[predicate] function that takes the index of an element and the element itself\n * and returns the result of predicate
evaluation on the element.\n * \n * @sample samples.collections.Collections.Filtering.filterIndexed\n */\npublic
inline fun BooleanArray.filterIndexed(predicate: (index: Int, Boolean) -> Boolean): List<Boolean> {\n    return
filterIndexedTo(ArrayList<Boolean>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching
the given [predicate].\n * @param [predicate] function that takes the index of an element and the element itself\n *
and returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexed\n */\npublic inline fun CharArray.filterIndexed(predicate:
(index: Int, Char) -> Boolean): List<Char> {\n    return filterIndexedTo(ArrayList<Char>(), predicate)\n}\n\n/**\n *
Appends all elements matching the given [predicate] to the given [destination].\n * @param [predicate] function that
takes the index of an element and the element itself\n * and returns the result of predicate evaluation on the
element.\n * \n * @sample samples.collections.Collections.Filtering.filterIndexedTo\n */\npublic inline fun <T, C :
MutableCollection<in T>> Array<out T>.filterIndexedTo(destination: C, predicate: (index: Int, T) -> Boolean): C
{\n    forEachIndexed { index, element ->\n        if (predicate(index, element)) destination.add(element)\n    }\n    return destination\n}\n\n/**\n * Appends all elements matching the given [predicate] to the given [destination].\n *
@param [predicate] function that takes the index of an element and the element itself\n * and returns the result of
predicate evaluation on the element.\n * \n * @sample samples.collections.Collections.Filtering.filterIndexedTo\n */\npublic inline fun <C : MutableCollection<in Byte>> ByteArray.filterIndexedTo(destination: C, predicate:
(index: Int, Byte) -> Boolean): C {\n    forEachIndexed { index, element ->\n        if (predicate(index, element))
destination.add(element)\n    }\n    return destination\n}\n\n/**\n * Appends all elements matching the given
[predicate] to the given [destination].\n * @param [predicate] function that takes the index of an element and the
element itself\n * and returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexedTo\n */\npublic inline fun <C : MutableCollection<in Short>>
ShortArray.filterIndexedTo(destination: C, predicate: (index: Int, Short) -> Boolean): C {\n    forEachIndexed {
index, element ->\n        if (predicate(index, element)) destination.add(element)\n    }\n    return
destination\n}\n\n/**\n * Appends all elements matching the given [predicate] to the given [destination].\n *

```

```

@param [predicate] function that takes the index of an element and the element itself\n * and returns the result of
predicate evaluation on the element.\n * \n * @sample samples.collections.Collections.Filtering.filterIndexedTo\n
*/\npublic inline fun <C : MutableCollection<in Int>> IntArray.filterIndexedTo(destination: C, predicate: (index:
Int, Int) -> Boolean): C {\n    forEachIndexed { index, element ->\n        if (predicate(index, element))
destination.add(element)\n    }\n    return destination\n}\n\n/**\n * Appends all elements matching the given
[predicate] to the given [destination].\n * @param [predicate] function that takes the index of an element and the
element itself\n * and returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexedTo\n
*/\npublic inline fun <C : MutableCollection<in Long>>
LongArray.filterIndexedTo(destination: C, predicate: (index: Int, Long) -> Boolean): C {\n    forEachIndexed {
index, element ->\n        if (predicate(index, element)) destination.add(element)\n    }\n    return
destination\n}\n\n/**\n * Appends all elements matching the given [predicate] to the given [destination].\n *
@param [predicate] function that takes the index of an element and the element itself\n * and returns the result of
predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexedTo\n
*/\npublic inline fun <C : MutableCollection<in Float>>
FloatArray.filterIndexedTo(destination: C, predicate:
(index: Int, Float) -> Boolean): C {\n    forEachIndexed { index, element ->\n        if (predicate(index, element))
destination.add(element)\n    }\n    return destination\n}\n\n/**\n * Appends all elements matching the given
[predicate] to the given [destination].\n * @param [predicate] function that takes the index of an element and the
element itself\n * and returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexedTo\n
*/\npublic inline fun <C : MutableCollection<in
Double>> DoubleArray.filterIndexedTo(destination: C, predicate: (index: Int, Double) -> Boolean): C {\n
    forEachIndexed { index, element ->\n        if (predicate(index, element)) destination.add(element)\n    }\n
    return destination\n}\n\n/**\n * Appends all elements matching the given [predicate] to the given [destination].\n *
@param [predicate] function that takes the index of an element and the element itself\n * and returns the result of
predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexedTo\n
*/\npublic inline fun <C : MutableCollection<in Boolean>>
BooleanArray.filterIndexedTo(destination: C, predicate:
(index: Int, Boolean) -> Boolean): C {\n    forEachIndexed { index, element ->\n        if (predicate(index, element))
destination.add(element)\n    }\n    return destination\n}\n\n/**\n * Appends all elements matching the given
[predicate] to the given [destination].\n * @param [predicate] function that takes the index of an element and the
element itself\n * and returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexedTo\n
*/\npublic inline fun <C : MutableCollection<in Char>>
CharArray.filterIndexedTo(destination: C, predicate: (index: Int, Char) -> Boolean): C {\n    forEachIndexed {
index, element ->\n        if (predicate(index, element)) destination.add(element)\n    }\n    return
destination\n}\n\n/**\n * Returns a list containing all elements that are instances of specified type parameter R.\n *
\n * @sample samples.collections.Collections.Filtering.filterIsInstance\n
*/\npublic inline fun <reified R>
Array<*>.filterIsInstance(): List<@kotlin.internal.NoInfer R> {\n    return
filterIsInstanceTo(ArrayList<R>())\n}\n\n/**\n * Appends all elements that are instances of specified type
parameter R to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterIsInstanceTo\n
*/\npublic inline fun <reified R, C :
MutableCollection<in R>> Array<*>.filterIsInstanceTo(destination: C): C {\n    for (element in this) if (element is
R) destination.add(element)\n    return destination\n}\n\n/**\n * Returns a list containing all elements not matching
the given [predicate].\n * \n * @sample samples.collections.Collections.Filtering.filter\n
*/\npublic inline fun <T>
Array<out T>.filterNot(predicate: (T) -> Boolean): List<T> {\n    return filterNotTo(ArrayList<T>(),
predicate)\n}\n\n/**\n * Returns a list containing all elements not matching the given [predicate].\n * \n * @sample
samples.collections.Collections.Filtering.filter\n
*/\npublic inline fun ByteArray.filterNot(predicate: (Byte) ->
Boolean): List<Byte> {\n    return filterNotTo(ArrayList<Byte>(), predicate)\n}\n\n/**\n * Returns a list containing
all elements not matching the given [predicate].\n * \n * @sample samples.collections.Collections.Filtering.filter\n
*/\npublic inline fun ShortArray.filterNot(predicate: (Short) -> Boolean): List<Short> {\n    return
filterNotTo(ArrayList<Short>(), predicate)\n}\n\n/**\n * Returns a list containing all elements not matching the

```

```

given [predicate].\n * \n * @sample samples.collections.Collections.Filtering.filter\n *^\npublic inline fun
IntArray.filterNot(predicate: (Int) -> Boolean): List<Int> {\n    return filterNotTo(ArrayList<Int>(),
predicate)\n}\n\n/**\n * Returns a list containing all elements not matching the given [predicate].\n * \n * @sample
samples.collections.Collections.Filtering.filter\n *^\npublic inline fun LongArray.filterNot(predicate: (Long) ->
Boolean): List<Long> {\n    return filterNotTo(ArrayList<Long>(), predicate)\n}\n\n/**\n * Returns a list
containing all elements not matching the given [predicate].\n * \n * @sample
samples.collections.Collections.Filtering.filter\n *^\npublic inline fun FloatArray.filterNot(predicate: (Float) ->
Boolean): List<Float> {\n    return filterNotTo(ArrayList<Float>(), predicate)\n}\n\n/**\n * Returns a list
containing all elements not matching the given [predicate].\n * \n * @sample
samples.collections.Collections.Filtering.filter\n *^\npublic inline fun DoubleArray.filterNot(predicate: (Double) ->
Boolean): List<Double> {\n    return filterNotTo(ArrayList<Double>(), predicate)\n}\n\n/**\n * Returns a list
containing all elements not matching the given [predicate].\n * \n * @sample
samples.collections.Collections.Filtering.filter\n *^\npublic inline fun BooleanArray.filterNot(predicate: (Boolean) -
> Boolean): List<Boolean> {\n    return filterNotTo(ArrayList<Boolean>(), predicate)\n}\n\n/**\n * Returns a list
containing all elements not matching the given [predicate].\n * \n * @sample
samples.collections.Collections.Filtering.filter\n *^\npublic inline fun CharArray.filterNot(predicate: (Char) ->
Boolean): List<Char> {\n    return filterNotTo(ArrayList<Char>(), predicate)\n}\n\n/**\n * Returns a list containing
all elements that are not `null`.\n * \n * @sample samples.collections.Collections.Filtering.filterNotNull\n *^\npublic
fun <T : Any> Array<out T?>.filterNotNull(): List<T> {\n    return filterNotNullTo(ArrayList<T>())\n}\n\n/**\n *
Appends all elements that are not `null` to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterNotNullTo\n *^\npublic fun <C : MutableCollection<in T>, T : Any>
Array<out T?>.filterNotNullTo(destination: C): C {\n    for (element in this) if (element != null)
destination.add(element)\n    return destination\n}\n\n/**\n * Appends all elements not matching the given
[predicate] to the given [destination].\n * \n * @sample samples.collections.Collections.Filtering.filterTo\n *^\npublic
inline fun <T, C : MutableCollection<in T>> Array<out T>.filterNotTo(destination: C, predicate: (T) ->
Boolean): C {\n    for (element in this) if (!predicate(element)) destination.add(element)\n    return
destination\n}\n\n/**\n * Appends all elements not matching the given [predicate] to the given [destination].\n * \n *
@sample samples.collections.Collections.Filtering.filterTo\n *^\npublic inline fun <C : MutableCollection<in
Byte>> ByteArray.filterNotTo(destination: C, predicate: (Byte) -> Boolean): C {\n    for (element in this) if
(!predicate(element)) destination.add(element)\n    return destination\n}\n\n/**\n * Appends all elements not
matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n *^\npublic inline fun <C : MutableCollection<in Short>>
ShortArray.filterNotTo(destination: C, predicate: (Short) -> Boolean): C {\n    for (element in this) if
(!predicate(element)) destination.add(element)\n    return destination\n}\n\n/**\n * Appends all elements not
matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n *^\npublic inline fun <C : MutableCollection<in Int>>
IntArray.filterNotTo(destination: C, predicate: (Int) -> Boolean): C {\n    for (element in this) if
(!predicate(element)) destination.add(element)\n    return destination\n}\n\n/**\n * Appends all elements not
matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n *^\npublic inline fun <C : MutableCollection<in Long>>
LongArray.filterNotTo(destination: C, predicate: (Long) -> Boolean): C {\n    for (element in this) if
(!predicate(element)) destination.add(element)\n    return destination\n}\n\n/**\n * Appends all elements not
matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n *^\npublic inline fun <C : MutableCollection<in Float>>
FloatArray.filterNotTo(destination: C, predicate: (Float) -> Boolean): C {\n    for (element in this) if
(!predicate(element)) destination.add(element)\n    return destination\n}\n\n/**\n * Appends all elements not
matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n *^\npublic inline fun <C : MutableCollection<in Double>>

```

```

DoubleArray.filterNotTo(destination: C, predicate: (Double) -> Boolean): C {
    for (element in this) if (!predicate(element)) destination.add(element)
}
return destination
}
Append all elements not matching the given [predicate] to the given [destination].
@sample
samples.collections.Collections.Filtering.filterTo
public inline fun <C : MutableCollection<in Boolean>>
BooleanArray.filterNotTo(destination: C, predicate: (Boolean) -> Boolean): C {
    for (element in this) if (!predicate(element)) destination.add(element)
}
return destination
}
Append all elements not matching the given [predicate] to the given [destination].
@sample
samples.collections.Collections.Filtering.filterTo
public inline fun <C : MutableCollection<in Char>>
CharArray.filterNotTo(destination: C, predicate: (Char) -> Boolean): C {
    for (element in this) if (!predicate(element)) destination.add(element)
}
return destination
}
Append all elements matching the given [predicate] to the given [destination].
@sample
samples.collections.Collections.Filtering.filterTo
public inline fun <T, C : MutableCollection<in T>> Array<out T>.filterTo(destination: C, predicate: (T) -> Boolean): C {
    for (element in this) if (predicate(element)) destination.add(element)
}
return destination
}
Append all elements matching the given [predicate] to the given [destination].
@sample
samples.collections.Collections.Filtering.filterTo
public inline fun <C : MutableCollection<in Byte>> ByteArray.filterTo(destination: C, predicate: (Byte) -> Boolean): C {
    for (element in this) if (predicate(element)) destination.add(element)
}
return destination
}
Append all elements matching the given [predicate] to the given [destination].
@sample
samples.collections.Collections.Filtering.filterTo
public inline fun <C : MutableCollection<in Short>> ShortArray.filterTo(destination: C, predicate: (Short) -> Boolean): C {
    for (element in this) if (predicate(element)) destination.add(element)
}
return destination
}
Append all elements matching the given [predicate] to the given [destination].
@sample
samples.collections.Collections.Filtering.filterTo
public inline fun <C : MutableCollection<in Int>> IntArray.filterTo(destination: C, predicate: (Int) -> Boolean): C {
    for (element in this) if (predicate(element)) destination.add(element)
}
return destination
}
Append all elements matching the given [predicate] to the given [destination].
@sample
samples.collections.Collections.Filtering.filterTo
public inline fun <C : MutableCollection<in Long>> LongArray.filterTo(destination: C, predicate: (Long) -> Boolean): C {
    for (element in this) if (predicate(element)) destination.add(element)
}
return destination
}
Append all elements matching the given [predicate] to the given [destination].
@sample
samples.collections.Collections.Filtering.filterTo
public inline fun <C : MutableCollection<in Float>> FloatArray.filterTo(destination: C, predicate: (Float) -> Boolean): C {
    for (element in this) if (predicate(element)) destination.add(element)
}
return destination
}
Append all elements matching the given [predicate] to the given [destination].
@sample
samples.collections.Collections.Filtering.filterTo
public inline fun <C : MutableCollection<in Double>> DoubleArray.filterTo(destination: C, predicate: (Double) -> Boolean): C {
    for (element in this) if (predicate(element)) destination.add(element)
}
return destination
}
Append all elements matching the given [predicate] to the given [destination].
@sample
samples.collections.Collections.Filtering.filterTo
public inline fun <C : MutableCollection<in Boolean>> BooleanArray.filterTo(destination: C, predicate: (Boolean) -> Boolean): C {
    for (element in this) if (predicate(element)) destination.add(element)
}
return destination
}
Append all elements matching the given [predicate] to the given [destination].
@sample
samples.collections.Collections.Filtering.filterTo
public inline fun <C : MutableCollection<in Char>> CharArray.filterTo(destination: C, predicate: (Char) -> Boolean): C {
    for (element in this) if (predicate(element)) destination.add(element)
}
return destination
}
Returns a list containing elements at indices in the specified [indices] range.
public fun <T> Array<out T>.slice(indices: IntRange): List<T> {
    if (indices.isEmpty()) return listOf()
}
return copyOfRange(indices.start, indices.endInclusive + 1).asList()
}
Returns a list containing elements at indices in the specified [indices] range.
public fun ByteArray.slice(indices: IntRange): List<Byte> {
    if (indices.isEmpty()) return listOf()
}
return copyOfRange(indices.start, indices.endInclusive + 1).asList()
}
Returns a list containing elements at indices in the specified [indices] range.
public fun ShortArray.slice(indices: IntRange): List<Short> {
    if (indices.isEmpty()) return listOf()
}
return

```

```

copyOfRange(indices.start, indices.endInclusive + 1).asList()\n\n/**\n * Returns a list containing elements at
indices in the specified [indices] range.\n */\npublic fun IntArray.slice(indices: IntRange): List<Int> {\n    if
(indices.isEmpty()) return listOf()\n    return copyOfRange(indices.start, indices.endInclusive +
1).asList()\n}\n\n/**\n * Returns a list containing elements at indices in the specified [indices] range.\n */\npublic
fun LongArray.slice(indices: IntRange): List<Long> {\n    if (indices.isEmpty()) return listOf()\n    return
copyOfRange(indices.start, indices.endInclusive + 1).asList()\n}\n\n/**\n * Returns a list containing elements at
indices in the specified [indices] range.\n */\npublic fun FloatArray.slice(indices: IntRange): List<Float> {\n    if
(indices.isEmpty()) return listOf()\n    return copyOfRange(indices.start, indices.endInclusive +
1).asList()\n}\n\n/**\n * Returns a list containing elements at indices in the specified [indices] range.\n */\npublic
fun DoubleArray.slice(indices: IntRange): List<Double> {\n    if (indices.isEmpty()) return listOf()\n    return
copyOfRange(indices.start, indices.endInclusive + 1).asList()\n}\n\n/**\n * Returns a list containing elements at
indices in the specified [indices] range.\n */\npublic fun BooleanArray.slice(indices: IntRange): List<Boolean> {\n
if (indices.isEmpty()) return listOf()\n    return copyOfRange(indices.start, indices.endInclusive +
1).asList()\n}\n\n/**\n * Returns a list containing elements at indices in the specified [indices] range.\n */\npublic
fun CharArray.slice(indices: IntRange): List<Char> {\n    if (indices.isEmpty()) return listOf()\n    return
copyOfRange(indices.start, indices.endInclusive + 1).asList()\n}\n\n/**\n * Returns a list containing elements at
specified [indices].\n */\npublic fun <T> Array<out T>.slice(indices: Iterable<Int>): List<T> {\n    val size =
indices.collectionSizeOrDefault(10)\n    if (size == 0) return emptyList()\n    val list = ArrayList<T>(size)\n    for
(index in indices) {\n        list.add(get(index))\n    }\n    return list\n}\n\n/**\n * Returns a list containing elements at
specified [indices].\n */\npublic fun ByteArray.slice(indices: Iterable<Int>): List<Byte> {\n    val size =
indices.collectionSizeOrDefault(10)\n    if (size == 0) return emptyList()\n    val list = ArrayList<Byte>(size)\n    for
(index in indices) {\n        list.add(get(index))\n    }\n    return list\n}\n\n/**\n * Returns a list containing elements at
specified [indices].\n */\npublic fun ShortArray.slice(indices: Iterable<Int>): List<Short> {\n    val size =
indices.collectionSizeOrDefault(10)\n    if (size == 0) return emptyList()\n    val list = ArrayList<Short>(size)\n
for (index in indices) {\n        list.add(get(index))\n    }\n    return list\n}\n\n/**\n * Returns a list containing
elements at specified [indices].\n */\npublic fun IntArray.slice(indices: Iterable<Int>): List<Int> {\n    val size =
indices.collectionSizeOrDefault(10)\n    if (size == 0) return emptyList()\n    val list = ArrayList<Int>(size)\n    for
(index in indices) {\n        list.add(get(index))\n    }\n    return list\n}\n\n/**\n * Returns a list containing elements at
specified [indices].\n */\npublic fun LongArray.slice(indices: Iterable<Int>): List<Long> {\n    val size =
indices.collectionSizeOrDefault(10)\n    if (size == 0) return emptyList()\n    val list = ArrayList<Long>(size)\n
for (index in indices) {\n        list.add(get(index))\n    }\n    return list\n}\n\n/**\n * Returns a list containing
elements at specified [indices].\n */\npublic fun FloatArray.slice(indices: Iterable<Int>): List<Float> {\n    val size =
indices.collectionSizeOrDefault(10)\n    if (size == 0) return emptyList()\n    val list = ArrayList<Float>(size)\n
for (index in indices) {\n        list.add(get(index))\n    }\n    return list\n}\n\n/**\n * Returns a list containing
elements at specified [indices].\n */\npublic fun DoubleArray.slice(indices: Iterable<Int>): List<Double> {\n    val
size = indices.collectionSizeOrDefault(10)\n    if (size == 0) return emptyList()\n    val list =
ArrayList<Double>(size)\n    for (index in indices) {\n        list.add(get(index))\n    }\n    return list\n}\n\n/**\n *
Returns a list containing elements at specified [indices].\n */\npublic fun BooleanArray.slice(indices: Iterable<Int>):
List<Boolean> {\n    val size = indices.collectionSizeOrDefault(10)\n    if (size == 0) return emptyList()\n    val list
= ArrayList<Boolean>(size)\n    for (index in indices) {\n        list.add(get(index))\n    }\n    return list\n}\n\n/**\n *
Returns a list containing elements at specified [indices].\n */\npublic fun CharArray.slice(indices: Iterable<Int>):
List<Char> {\n    val size = indices.collectionSizeOrDefault(10)\n    if (size == 0) return emptyList()\n    val list =
ArrayList<Char>(size)\n    for (index in indices) {\n        list.add(get(index))\n    }\n    return list\n}\n\n/**\n *
Returns an array containing elements of this array at specified [indices].\n */\npublic fun <T>
Array<T>.sliceArray(indices: Collection<Int>): Array<T> {\n    val result = arrayOfNulls(this, indices.size)\n    var
targetIndex = 0\n    for (sourceIndex in indices) {\n        result[targetIndex++] = this[sourceIndex]\n    }\n    return
result\n}\n\n/**\n * Returns an array containing elements of this array at specified [indices].\n */\npublic fun
ByteArray.sliceArray(indices: Collection<Int>): ByteArray {\n    val result = ByteArray(indices.size)\n    var

```

```

targetIndex = 0\n  for (sourceIndex in indices) {\n    result[targetIndex++] = this[sourceIndex]\n  }\n  return
result\n}\n\n/**\n * Returns an array containing elements of this array at specified [indices].\n *\npublic fun
ShortArray.sliceArray(indices: Collection<Int>): ShortArray {\n  val result = ShortArray(indices.size)\n  var
targetIndex = 0\n  for (sourceIndex in indices) {\n    result[targetIndex++] = this[sourceIndex]\n  }\n  return
result\n}\n\n/**\n * Returns an array containing elements of this array at specified [indices].\n *\npublic fun
IntArray.sliceArray(indices: Collection<Int>): IntArray {\n  val result = IntArray(indices.size)\n  var targetIndex
= 0\n  for (sourceIndex in indices) {\n    result[targetIndex++] = this[sourceIndex]\n  }\n  return
result\n}\n\n/**\n * Returns an array containing elements of this array at specified [indices].\n *\npublic fun
LongArray.sliceArray(indices: Collection<Int>): LongArray {\n  val result = LongArray(indices.size)\n  var
targetIndex = 0\n  for (sourceIndex in indices) {\n    result[targetIndex++] = this[sourceIndex]\n  }\n  return
result\n}\n\n/**\n * Returns an array containing elements of this array at specified [indices].\n *\npublic fun
FloatArray.sliceArray(indices: Collection<Int>): FloatArray {\n  val result = FloatArray(indices.size)\n  var
targetIndex = 0\n  for (sourceIndex in indices) {\n    result[targetIndex++] = this[sourceIndex]\n  }\n  return
result\n}\n\n/**\n * Returns an array containing elements of this array at specified [indices].\n *\npublic fun
DoubleArray.sliceArray(indices: Collection<Int>): DoubleArray {\n  val result = DoubleArray(indices.size)\n
var targetIndex = 0\n  for (sourceIndex in indices) {\n    result[targetIndex++] = this[sourceIndex]\n  }\n
return result\n}\n\n/**\n * Returns an array containing elements of this array at specified [indices].\n *\npublic fun
BooleanArray.sliceArray(indices: Collection<Int>): BooleanArray {\n  val result = BooleanArray(indices.size)\n
var targetIndex = 0\n  for (sourceIndex in indices) {\n    result[targetIndex++] = this[sourceIndex]\n  }\n
return result\n}\n\n/**\n * Returns an array containing elements of this array at specified [indices].\n *\npublic fun
CharArray.sliceArray(indices: Collection<Int>): CharArray {\n  val result = CharArray(indices.size)\n  var
targetIndex = 0\n  for (sourceIndex in indices) {\n    result[targetIndex++] = this[sourceIndex]\n  }\n  return
result\n}\n\n/**\n * Returns an array containing elements at indices in the specified [indices] range.\n *\npublic fun
<T> Array<T>.sliceArray(indices: IntRange): Array<T> {\n  if (indices.isEmpty()) return copyOfRange(0, 0)\n
return copyOfRange(indices.start, indices.endInclusive + 1)\n}\n\n/**\n * Returns an array containing elements at
indices in the specified [indices] range.\n *\npublic fun ByteArray.sliceArray(indices: IntRange): ByteArray {\n  if
(indices.isEmpty()) return ByteArray(0)\n  return copyOfRange(indices.start, indices.endInclusive + 1)\n}\n\n/**\n
* Returns an array containing elements at indices in the specified [indices] range.\n *\npublic fun
ShortArray.sliceArray(indices: IntRange): ShortArray {\n  if (indices.isEmpty()) return ShortArray(0)\n  return
copyOfRange(indices.start, indices.endInclusive + 1)\n}\n\n/**\n * Returns an array containing elements at indices
in the specified [indices] range.\n *\npublic fun IntArray.sliceArray(indices: IntRange): IntArray {\n  if
(indices.isEmpty()) return IntArray(0)\n  return copyOfRange(indices.start, indices.endInclusive + 1)\n}\n\n/**\n
* Returns an array containing elements at indices in the specified [indices] range.\n *\npublic fun
LongArray.sliceArray(indices: IntRange): LongArray {\n  if (indices.isEmpty()) return LongArray(0)\n  return
copyOfRange(indices.start, indices.endInclusive + 1)\n}\n\n/**\n * Returns an array containing elements at indices
in the specified [indices] range.\n *\npublic fun FloatArray.sliceArray(indices: IntRange): FloatArray {\n  if
(indices.isEmpty()) return FloatArray(0)\n  return copyOfRange(indices.start, indices.endInclusive + 1)\n}\n\n/**\n
* Returns an array containing elements at indices in the specified [indices] range.\n *\npublic fun
DoubleArray.sliceArray(indices: IntRange): DoubleArray {\n  if (indices.isEmpty()) return DoubleArray(0)\n
return copyOfRange(indices.start, indices.endInclusive + 1)\n}\n\n/**\n * Returns an array containing elements at
indices in the specified [indices] range.\n *\npublic fun BooleanArray.sliceArray(indices: IntRange): BooleanArray
{\n  if (indices.isEmpty()) return BooleanArray(0)\n  return copyOfRange(indices.start, indices.endInclusive +
1)\n}\n\n/**\n * Returns an array containing elements at indices in the specified [indices] range.\n *\npublic fun
CharArray.sliceArray(indices: IntRange): CharArray {\n  if (indices.isEmpty()) return CharArray(0)\n  return
copyOfRange(indices.start, indices.endInclusive + 1)\n}\n\n/**\n * Returns a list containing first [n] elements.\n *\n
* @throws IllegalArgumentException if [n] is negative.\n *\n * @sample
samples.collections.Collections.Transformations.take\n *\npublic fun <T> Array<out T>.take(n: Int): List<T> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  if (n == 0) return emptyList()\n  if (n >=

```

```

size) return toList()\n if (n == 1) return listOf(this[0])\n var count = 0\n val list = ArrayList<T>(n)\n for
(item in this) {\n list.add(item)\n if (++count == n)\n break\n }\n return list\n}\n\n/**\n * Returns
a list containing first [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n */\npublic fun ByteArray.take(n: Int): List<Byte> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n if (n >=
size) return toList()\n if (n == 1) return listOf(this[0])\n var count = 0\n val list = ArrayList<Byte>(n)\n for
(item in this) {\n list.add(item)\n if (++count == n)\n break\n }\n return list\n}\n\n/**\n * Returns
a list containing first [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n */\npublic fun ShortArray.take(n: Int): List<Short> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n if (n >=
size) return toList()\n if (n == 1) return listOf(this[0])\n var count = 0\n val list = ArrayList<Short>(n)\n for
(item in this) {\n list.add(item)\n if (++count == n)\n break\n }\n return list\n}\n\n/**\n * Returns
a list containing first [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n */\npublic fun IntArray.take(n: Int): List<Int> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n if (n >=
size) return toList()\n if (n == 1) return listOf(this[0])\n var count = 0\n val list = ArrayList<Int>(n)\n for
(item in this) {\n list.add(item)\n if (++count == n)\n break\n }\n return list\n}\n\n/**\n * Returns
a list containing first [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n */\npublic fun LongArray.take(n: Int): List<Long> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n if (n >=
size) return toList()\n if (n == 1) return listOf(this[0])\n var count = 0\n val list = ArrayList<Long>(n)\n for
(item in this) {\n list.add(item)\n if (++count == n)\n break\n }\n return list\n}\n\n/**\n * Returns
a list containing first [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n */\npublic fun FloatArray.take(n: Int): List<Float> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n if (n >=
size) return toList()\n if (n == 1) return listOf(this[0])\n var count = 0\n val list = ArrayList<Float>(n)\n for
(item in this) {\n list.add(item)\n if (++count == n)\n break\n }\n return list\n}\n\n/**\n * Returns
a list containing first [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n */\npublic fun DoubleArray.take(n: Int): List<Double> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n if (n >=
size) return toList()\n if (n == 1) return listOf(this[0])\n var count = 0\n val list = ArrayList<Double>(n)\n
for (item in this) {\n list.add(item)\n if (++count == n)\n break\n }\n return list\n}\n\n/**\n *
Returns a list containing first [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n *
@sample samples.collections.Collections.Transformations.take\n */\npublic fun BooleanArray.take(n: Int):
List<Boolean> {\n require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return
emptyList()\n if (n >= size) return toList()\n if (n == 1) return listOf(this[0])\n var count = 0\n val list =
ArrayList<Boolean>(n)\n for (item in this) {\n list.add(item)\n if (++count == n)\n break\n }\n
return list\n}\n\n/**\n * Returns a list containing first [n] elements.\n * \n * @throws IllegalArgumentException if
[n] is negative.\n * \n * @sample samples.collections.Collections.Transformations.take\n */\npublic fun
CharArray.take(n: Int): List<Char> {\n require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if
(n == 0) return emptyList()\n if (n >= size) return toList()\n if (n == 1) return listOf(this[0])\n var count = 0\n
val list = ArrayList<Char>(n)\n for (item in this) {\n list.add(item)\n if (++count == n)\n break\n
}\n return list\n}\n\n/**\n * Returns a list containing last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n */\npublic fun <T> Array<out T>.takeLast(n: Int): List<T>
{\n require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n val
size = size\n if (n >= size) return toList()\n if (n == 1) return listOf(this[size - 1])\n val list =
ArrayList<T>(n)\n for (index in size - n until size)\n list.add(this[index])\n return list\n}\n\n/**\n * Returns

```

```

a list containing last [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n * \npublic fun ByteArray.takeLast(n: Int): List<Byte> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n val size =
size\n if (n >= size) return toList()\n if (n == 1) return listOf(this[size - 1])\n val list = ArrayList<Byte>(n)\n
for (index in size - n until size)\n list.add(this[index])\n return list\n}\n\n/**\n * Returns a list containing last
[n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n * \npublic fun ShortArray.takeLast(n: Int): List<Short> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n val size =
size\n if (n >= size) return toList()\n if (n == 1) return listOf(this[size - 1])\n val list = ArrayList<Short>(n)\n
for (index in size - n until size)\n list.add(this[index])\n return list\n}\n\n/**\n * Returns a list containing last
[n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n * \npublic fun IntArray.takeLast(n: Int): List<Int> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n val size =
size\n if (n >= size) return toList()\n if (n == 1) return listOf(this[size - 1])\n val list = ArrayList<Int>(n)\n
for (index in size - n until size)\n list.add(this[index])\n return list\n}\n\n/**\n * Returns a list containing last
[n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n * \npublic fun LongArray.takeLast(n: Int): List<Long> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n val size =
size\n if (n >= size) return toList()\n if (n == 1) return listOf(this[size - 1])\n val list = ArrayList<Long>(n)\n
for (index in size - n until size)\n list.add(this[index])\n return list\n}\n\n/**\n * Returns a list containing last
[n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n * \npublic fun FloatArray.takeLast(n: Int): List<Float> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n val size =
size\n if (n >= size) return toList()\n if (n == 1) return listOf(this[size - 1])\n val list = ArrayList<Float>(n)\n
for (index in size - n until size)\n list.add(this[index])\n return list\n}\n\n/**\n * Returns a list containing last
[n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n * \npublic fun DoubleArray.takeLast(n: Int): List<Double>
{\n require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return emptyList()\n val
size = size\n if (n >= size) return toList()\n if (n == 1) return listOf(this[size - 1])\n val list =
ArrayList<Double>(n)\n for (index in size - n until size)\n list.add(this[index])\n return list\n}\n\n/**\n *
Returns a list containing last [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n *
@sample samples.collections.Collections.Transformations.take\n * \npublic fun BooleanArray.takeLast(n: Int):
List<Boolean> {\n require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return
emptyList()\n val size = size\n if (n >= size) return toList()\n if (n == 1) return listOf(this[size - 1])\n val list
= ArrayList<Boolean>(n)\n for (index in size - n until size)\n list.add(this[index])\n return list\n}\n\n/**\n *
Returns a list containing last [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n *
@sample samples.collections.Collections.Transformations.take\n * \npublic fun CharArray.takeLast(n: Int):
List<Char> {\n require(n >= 0) { \"Requested element count $n is less than zero.\" }\n if (n == 0) return
emptyList()\n val size = size\n if (n >= size) return toList()\n if (n == 1) return listOf(this[size - 1])\n val list
= ArrayList<Char>(n)\n for (index in size - n until size)\n list.add(this[index])\n return list\n}\n\n/**\n *
Returns a list containing last elements satisfying the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.take\n * \npublic inline fun <T> Array<out
T>.takeLastWhile(predicate: (T) -> Boolean): List<T> {\n for (index in lastIndex downTo 0) {\n if
(!predicate(this[index])) {\n return drop(index + 1)\n }\n }\n return toList()\n}\n\n/**\n * Returns a
list containing last elements satisfying the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.take\n * \npublic inline fun ByteArray.takeLastWhile(predicate:
(Byte) -> Boolean): List<Byte> {\n for (index in lastIndex downTo 0) {\n if (!predicate(this[index])) {\n
return drop(index + 1)\n }\n }\n return toList()\n}\n\n/**\n * Returns a list containing last elements

```

```

satisfying the given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n *^\npublic
inline fun ShortArray.takeLastWhile(predicate: (Short) -> Boolean): List<Short> {\n  for (index in lastIndex
downTo 0) {\n    if (!predicate(this[index])) {\n      return drop(index + 1)\n    }\n  }\n  return
toList()\n}\n\n/**\n * Returns a list containing last elements satisfying the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.take\n *^\npublic inline fun IntArray.takeLastWhile(predicate: (Int)
-> Boolean): List<Int> {\n  for (index in lastIndex downTo 0) {\n    if (!predicate(this[index])) {\n      return
drop(index + 1)\n    }\n  }\n  return toList()\n}\n\n/**\n * Returns a list containing last elements satisfying the
given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n *^\npublic inline fun
LongArray.takeLastWhile(predicate: (Long) -> Boolean): List<Long> {\n  for (index in lastIndex downTo 0) {\n
  if (!predicate(this[index])) {\n    return drop(index + 1)\n  }\n}\n  return toList()\n}\n\n/**\n * Returns
a list containing last elements satisfying the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.take\n *^\npublic inline fun FloatArray.takeLastWhile(predicate:
(Float) -> Boolean): List<Float> {\n  for (index in lastIndex downTo 0) {\n    if (!predicate(this[index])) {\n
  return drop(index + 1)\n    }\n  }\n  return toList()\n}\n\n/**\n * Returns a list containing last elements
satisfying the given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n *^\npublic
inline fun DoubleArray.takeLastWhile(predicate: (Double) -> Boolean): List<Double> {\n  for (index in lastIndex
downTo 0) {\n    if (!predicate(this[index])) {\n      return drop(index + 1)\n    }\n  }\n  return
toList()\n}\n\n/**\n * Returns a list containing last elements satisfying the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.take\n *^\npublic inline fun
BooleanArray.takeLastWhile(predicate: (Boolean) -> Boolean): List<Boolean> {\n  for (index in lastIndex
downTo 0) {\n    if (!predicate(this[index])) {\n      return drop(index + 1)\n    }\n  }\n  return
toList()\n}\n\n/**\n * Returns a list containing last elements satisfying the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.take\n *^\npublic inline fun CharArray.takeLastWhile(predicate:
(Char) -> Boolean): List<Char> {\n  for (index in lastIndex downTo 0) {\n    if (!predicate(this[index])) {\n
  return drop(index + 1)\n    }\n  }\n  return toList()\n}\n\n/**\n * Returns a list containing first elements
satisfying the given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n *^\npublic
inline fun <T> Array<out T>.takeWhile(predicate: (T) -> Boolean): List<T> {\n  val list = ArrayList<T>()\n  for
(item in this) {\n    if (!predicate(item))\n      break\n    list.add(item)\n  }\n  return list\n}\n\n/**\n *
Returns a list containing first elements satisfying the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.take\n *^\npublic inline fun ByteArray.takeWhile(predicate: (Byte)
-> Boolean): List<Byte> {\n  val list = ArrayList<Byte>()\n  for (item in this) {\n    if (!predicate(item))\n
  break\n    list.add(item)\n  }\n  return list\n}\n\n/**\n * Returns a list containing first elements satisfying the
given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n *^\npublic inline fun
ShortArray.takeWhile(predicate: (Short) -> Boolean): List<Short> {\n  val list = ArrayList<Short>()\n  for (item
in this) {\n    if (!predicate(item))\n      break\n    list.add(item)\n  }\n  return list\n}\n\n/**\n * Returns a
list containing first elements satisfying the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.take\n *^\npublic inline fun IntArray.takeWhile(predicate: (Int) ->
Boolean): List<Int> {\n  val list = ArrayList<Int>()\n  for (item in this) {\n    if (!predicate(item))\n
  break\n    list.add(item)\n  }\n  return list\n}\n\n/**\n * Returns a list containing first elements satisfying the
given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n *^\npublic inline fun
LongArray.takeWhile(predicate: (Long) -> Boolean): List<Long> {\n  val list = ArrayList<Long>()\n  for (item
in this) {\n    if (!predicate(item))\n      break\n    list.add(item)\n  }\n  return list\n}\n\n/**\n * Returns a
list containing first elements satisfying the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.take\n *^\npublic inline fun FloatArray.takeWhile(predicate:
(Float) -> Boolean): List<Float> {\n  val list = ArrayList<Float>()\n  for (item in this) {\n    if
(!predicate(item))\n      break\n    list.add(item)\n  }\n  return list\n}\n\n/**\n * Returns a list containing first
elements satisfying the given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n
*^\npublic inline fun DoubleArray.takeWhile(predicate: (Double) -> Boolean): List<Double> {\n  val list =

```

```

ArrayList<Double>()
for (item in this) {
    if (!predicate(item))
        break
    list.add(item)
}
return list
}
/**
 * Returns a list containing first elements satisfying the given [predicate].
 */
@sample
samples.collections.Collections.Transformations.take
*/
public inline fun BooleanArray.takeWhile(predicate:
(Boolean) -> Boolean): List<Boolean> {
    val list = ArrayList<Boolean>()
    for (item in this) {
        if (!predicate(item))
            break
        list.add(item)
    }
    return list
}
/**
 * Returns a list containing first
elements satisfying the given [predicate].
 */
@sample
samples.collections.Collections.Transformations.take
*/
public inline fun CharArray.takeWhile(predicate: (Char) -> Boolean): List<Char> {
    val list =
ArrayList<Char>()
for (item in this) {
    if (!predicate(item))
        break
    list.add(item)
}
return list
}
/**
 * Reverses elements in the array in-place.
 */
public fun <T> Array<T>.reverse(): Unit {
    val midPoint = (size / 2) - 1
    if (midPoint < 0) return
    var reverseIndex = lastIndex
    for (index in
0..midPoint) {
        val tmp = this[index]
        this[index] = this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}
/**
 * Reverses elements in the array in-place.
 */
public fun ByteArray.reverse():
Unit {
    val midPoint = (size / 2) - 1
    if (midPoint < 0) return
    var reverseIndex = lastIndex
    for (index in
0..midPoint) {
        val tmp = this[index]
        this[index] = this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}
/**
 * Reverses elements in the array in-place.
 */
public fun ShortArray.reverse():
Unit {
    val midPoint = (size / 2) - 1
    if (midPoint < 0) return
    var reverseIndex = lastIndex
    for (index in
0..midPoint) {
        val tmp = this[index]
        this[index] = this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}
/**
 * Reverses elements in the array in-place.
 */
public fun IntArray.reverse():
Unit {
    val midPoint = (size / 2) - 1
    if (midPoint < 0) return
    var reverseIndex = lastIndex
    for (index in
0..midPoint) {
        val tmp = this[index]
        this[index] = this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}
/**
 * Reverses elements in the array in-place.
 */
public fun LongArray.reverse():
Unit {
    val midPoint = (size / 2) - 1
    if (midPoint < 0) return
    var reverseIndex = lastIndex
    for (index in
0..midPoint) {
        val tmp = this[index]
        this[index] = this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}
/**
 * Reverses elements in the array in-place.
 */
public fun
DoubleArray.reverse(): Unit {
    val midPoint = (size / 2) - 1
    if (midPoint < 0) return
    var reverseIndex =
lastIndex
    for (index in 0..midPoint) {
        val tmp = this[index]
        this[index] = this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}
/**
 * Reverses elements in the array in-place.
 */
public fun BooleanArray.reverse(): Unit {
    val midPoint = (size / 2) - 1
    if (midPoint < 0) return
    var
reverseIndex = lastIndex
    for (index in 0..midPoint) {
        val tmp = this[index]
        this[index] =
this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}
/**
 * Reverses elements in the
array in-place.
 */
public fun CharArray.reverse(): Unit {
    val midPoint = (size / 2) - 1
    if (midPoint < 0)
return
    var reverseIndex = lastIndex
    for (index in 0..midPoint) {
        val tmp = this[index]
        this[index]
= this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}
/**
 * Reverses elements of
the array in the specified range in-place.
 */
@sample
@param fromIndex the start of the range (inclusive) to reverse.
@param toIndex the end of the range (exclusive) to reverse.
@throws IndexOutOfBoundsException if
[fromIndex] is less than zero or [toIndex] is greater than the size of this array.
@throws
IllegalArgumentException if [fromIndex] is greater than [toIndex].
*/
@SinceKotlin("1.4")
public fun <T>
Array<T>.reverse(fromIndex: Int, toIndex: Int): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex,
size)
    val midPoint = (fromIndex + toIndex) / 2
    if (fromIndex == midPoint) return
    var reverseIndex =
toIndex - 1
    for (index in fromIndex until midPoint) {
        val tmp = this[index]
        this[index] =
this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}
/**
 * Reverses elements of the
array in the specified range in-place.
 */
@sample
@param fromIndex the start of the range (inclusive) to reverse.
@param toIndex the end of the range (exclusive) to reverse.
@throws IndexOutOfBoundsException if
[fromIndex] is less than zero or [toIndex] is greater than the size of this array.
@throws
IllegalArgumentException if [fromIndex] is greater than [toIndex].
*/
@SinceKotlin("1.4")
public fun

```

```

ByteArray.reverse(fromIndex: Int, toIndex: Int): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex,
size)
    val midPoint = (fromIndex + toIndex) / 2
    if (fromIndex == midPoint) return
    var reverseIndex =
toIndex - 1
    for (index in fromIndex until midPoint) {
        val tmp = this[index]
        this[index] =
this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}

/**
 * Reverses elements of the
array in the specified range in-place.
 * @param fromIndex the start of the range (inclusive) to reverse.
 * @param toIndex the end of the range (exclusive) to reverse.
 * @throws IndexOutOfBoundsException if
[fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws
IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.4")
public fun
ShortArray.reverse(fromIndex: Int, toIndex: Int): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex,
size)
    val midPoint = (fromIndex + toIndex) / 2
    if (fromIndex == midPoint) return
    var reverseIndex =
toIndex - 1
    for (index in fromIndex until midPoint) {
        val tmp = this[index]
        this[index] =
this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}

/**
 * Reverses elements of the
array in the specified range in-place.
 * @param fromIndex the start of the range (inclusive) to reverse.
 * @param toIndex the end of the range (exclusive) to reverse.
 * @throws IndexOutOfBoundsException if
[fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws
IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.4")
public fun
IntArray.reverse(fromIndex: Int, toIndex: Int): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex,
size)
    val midPoint = (fromIndex + toIndex) / 2
    if (fromIndex == midPoint) return
    var reverseIndex =
toIndex - 1
    for (index in fromIndex until midPoint) {
        val tmp = this[index]
        this[index] =
this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}

/**
 * Reverses elements of the
array in the specified range in-place.
 * @param fromIndex the start of the range (inclusive) to reverse.
 * @param toIndex the end of the range (exclusive) to reverse.
 * @throws IndexOutOfBoundsException if
[fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws
IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.4")
public fun
LongArray.reverse(fromIndex: Int, toIndex: Int): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex,
size)
    val midPoint = (fromIndex + toIndex) / 2
    if (fromIndex == midPoint) return
    var reverseIndex =
toIndex - 1
    for (index in fromIndex until midPoint) {
        val tmp = this[index]
        this[index] =
this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}

/**
 * Reverses elements of the
array in the specified range in-place.
 * @param fromIndex the start of the range (inclusive) to reverse.
 * @param toIndex the end of the range (exclusive) to reverse.
 * @throws IndexOutOfBoundsException if
[fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws
IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.4")
public fun
FloatArray.reverse(fromIndex: Int, toIndex: Int): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex,
size)
    val midPoint = (fromIndex + toIndex) / 2
    if (fromIndex == midPoint) return
    var reverseIndex =
toIndex - 1
    for (index in fromIndex until midPoint) {
        val tmp = this[index]
        this[index] =
this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}

/**
 * Reverses elements of the
array in the specified range in-place.
 * @param fromIndex the start of the range (inclusive) to reverse.
 * @param toIndex the end of the range (exclusive) to reverse.
 * @throws IndexOutOfBoundsException if
[fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws
IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.4")
public fun
DoubleArray.reverse(fromIndex: Int, toIndex: Int): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex,
size)
    val midPoint = (fromIndex + toIndex) / 2
    if (fromIndex == midPoint) return
    var reverseIndex =
toIndex - 1
    for (index in fromIndex until midPoint) {
        val tmp = this[index]
        this[index] =
this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}

/**
 * Reverses elements of the
array in the specified range in-place.
 * @param fromIndex the start of the range (inclusive) to reverse.
 * @param toIndex the end of the range (exclusive) to reverse.
 * @throws IndexOutOfBoundsException if
[fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws
IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.4")
public fun

```

```

BooleanArray.reverse(fromIndex: Int, toIndex: Int): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)
    val midPoint = (fromIndex + toIndex) / 2
    if (fromIndex == midPoint) return
    var reverseIndex = toIndex - 1
    for (index in fromIndex until midPoint) {
        val tmp = this[index]
        this[index] = this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}

/** Reverses elements of the array in the specified range in-place.
 * @param fromIndex the start of the range (inclusive) to reverse.
 * @param toIndex the end of the range (exclusive) to reverse.
 * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.4")
public fun CharArray.reverse(fromIndex: Int, toIndex: Int): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)
    val midPoint = (fromIndex + toIndex) / 2
    if (fromIndex == midPoint) return
    var reverseIndex = toIndex - 1
    for (index in fromIndex until midPoint) {
        val tmp = this[index]
        this[index] = this[reverseIndex]
        this[reverseIndex] = tmp
        reverseIndex--
    }
}

/** Returns a list with elements in reversed order.
 */
public fun <T> Array<out T>.reversed(): List<T> {
    if (isEmpty()) return emptyList()
    val list = toMutableList()
    list.reverse()
    return list
}

/** Returns a list with elements in reversed order.
 */
public fun ByteArray.reversed(): List<Byte> {
    if (isEmpty()) return emptyList()
    val list = toMutableList()
    list.reverse()
    return list
}

/** Returns a list with elements in reversed order.
 */
public fun ShortArray.reversed(): List<Short> {
    if (isEmpty()) return emptyList()
    val list = toMutableList()
    list.reverse()
    return list
}

/** Returns a list with elements in reversed order.
 */
public fun IntArray.reversed(): List<Int> {
    if (isEmpty()) return emptyList()
    val list = toMutableList()
    list.reverse()
    return list
}

/** Returns a list with elements in reversed order.
 */
public fun LongArray.reversed(): List<Long> {
    if (isEmpty()) return emptyList()
    val list = toMutableList()
    list.reverse()
    return list
}

/** Returns a list with elements in reversed order.
 */
public fun FloatArray.reversed(): List<Float> {
    if (isEmpty()) return emptyList()
    val list = toMutableList()
    list.reverse()
    return list
}

/** Returns a list with elements in reversed order.
 */
public fun DoubleArray.reversed(): List<Double> {
    if (isEmpty()) return emptyList()
    val list = toMutableList()
    list.reverse()
    return list
}

/** Returns a list with elements in reversed order.
 */
public fun BooleanArray.reversed(): List<Boolean> {
    if (isEmpty()) return emptyList()
    val list = toMutableList()
    list.reverse()
    return list
}

/** Returns a list with elements in reversed order.
 */
public fun CharArray.reversed(): List<Char> {
    if (isEmpty()) return emptyList()
    val list = toMutableList()
    list.reverse()
    return list
}

/** Returns an array with elements of this array in reversed order.
 */
public fun <T> Array<T>.reversedArray(): Array<T> {
    if (isEmpty()) return this
    val result = arrayOfNulls(this, size)
    val lastIndex = lastIndexOf()
    for (i in 0..lastIndex)
        result[lastIndex - i] = this[i]
    return result
}

/** Returns an array with elements of this array in reversed order.
 */
public fun ByteArray.reversedArray(): ByteArray {
    if (isEmpty()) return this
    val result = ByteArray(size)
    val lastIndex = lastIndexOf()
    for (i in 0..lastIndex)
        result[lastIndex - i] = this[i]
    return result
}

/** Returns an array with elements of this array in reversed order.
 */
public fun ShortArray.reversedArray(): ShortArray {
    if (isEmpty()) return this
    val result = ShortArray(size)
    val lastIndex = lastIndexOf()
    for (i in 0..lastIndex)
        result[lastIndex - i] = this[i]
    return result
}

/** Returns an array with elements of this array in reversed order.
 */
public fun IntArray.reversedArray(): IntArray {
    if (isEmpty()) return this
    val result = IntArray(size)
    val lastIndex = lastIndexOf()
    for (i in 0..lastIndex)
        result[lastIndex - i] = this[i]
    return result
}

/** Returns an array with elements of this array in reversed order.
 */
public fun LongArray.reversedArray(): LongArray {
    if (isEmpty()) return this
    val result = LongArray(size)
    val lastIndex = lastIndexOf()
    for (i in 0..lastIndex)
        result[lastIndex - i] = this[i]
    return result
}

/** Returns an array with elements of this array in reversed order.
 */
public fun FloatArray.reversedArray(): FloatArray {
    if (isEmpty()) return this
    val result = FloatArray(size)
    val lastIndex = lastIndexOf()
    for (i in 0..lastIndex)
        result[lastIndex - i] = this[i]
    return result
}

/** Returns an array with elements of this array in reversed order.
 */
public fun DoubleArray.reversedArray(): DoubleArray {
    if (isEmpty()) return this
    val result = DoubleArray(size)
    val lastIndex = lastIndexOf()
    for (i in 0..lastIndex)
        result[lastIndex -

```

```

i] = this[i]\n    return result\n}\n\n/**\n * Returns an array with elements of this array in reversed order.\n */\npublic
fun BooleanArray.reversedArray(): BooleanArray {\n    if (isEmpty()) return this\n    val result =
BooleanArray(size)\n    val lastIndex = lastIndex\n    for (i in 0..lastIndex)\n        result[lastIndex - i] = this[i]\n    return result\n}\n\n/**\n * Returns an array with elements of this array in reversed order.\n */\npublic fun
CharArray.reversedArray(): CharArray {\n    if (isEmpty()) return this\n    val result = CharArray(size)\n    val
lastIndex = lastIndex\n    for (i in 0..lastIndex)\n        result[lastIndex - i] = this[i]\n    return result\n}\n\n/**\n *
Randomly shuffles elements in this array in-place.\n */\n@SinceKotlin("1.4")\npublic fun <T>
Array<T>.shuffle(): Unit {\n    shuffle(Random)\n}\n\n/**\n * Randomly shuffles elements in this array in-place.\n
*/\n@SinceKotlin("1.4")\npublic fun ByteArray.shuffle(): Unit {\n    shuffle(Random)\n}\n\n/**\n * Randomly
shuffles elements in this array in-place.\n */\n@SinceKotlin("1.4")\npublic fun ShortArray.shuffle(): Unit {\n
shuffle(Random)\n}\n\n/**\n * Randomly shuffles elements in this array in-place.\n
*/\n@SinceKotlin("1.4")\npublic fun IntArray.shuffle(): Unit {\n    shuffle(Random)\n}\n\n/**\n * Randomly
shuffles elements in this array in-place.\n */\n@SinceKotlin("1.4")\npublic fun LongArray.shuffle(): Unit {\n
shuffle(Random)\n}\n\n/**\n * Randomly shuffles elements in this array in-place.\n
*/\n@SinceKotlin("1.4")\npublic fun FloatArray.shuffle(): Unit {\n    shuffle(Random)\n}\n\n/**\n * Randomly
shuffles elements in this array in-place.\n */\n@SinceKotlin("1.4")\npublic fun DoubleArray.shuffle(): Unit {\n
shuffle(Random)\n}\n\n/**\n * Randomly shuffles elements in this array in-place.\n
*/\n@SinceKotlin("1.4")\npublic fun BooleanArray.shuffle(): Unit {\n    shuffle(Random)\n}\n\n/**\n * Randomly
shuffles elements in this array in-place using the specified [random]
instance as the source of randomness.\n */\n * See:
https://en.wikipedia.org/wiki/Fisher%20%93Yates\_shuffle#The\_modern\_algorithm\n
*/\n@SinceKotlin("1.4")\npublic fun <T> Array<T>.shuffle(random: Random): Unit {\n    for (i in lastIndex
downTo 1) {\n        val j = random.nextInt(i + 1)\n        val copy = this[i]\n        this[i] = this[j]\n        this[j] = copy\n
    }\n}\n\n/**\n * Randomly shuffles elements in this array in-place using the specified [random] instance as the
source of randomness.\n */\n * See:
https://en.wikipedia.org/wiki/Fisher%20%93Yates\_shuffle#The\_modern\_algorithm\n
*/\n@SinceKotlin("1.4")\npublic fun ByteArray.shuffle(random: Random): Unit {\n    for (i in lastIndex downTo
1) {\n        val j = random.nextInt(i + 1)\n        val copy = this[i]\n        this[i] = this[j]\n        this[j] = copy\n
    }\n}\n\n/**\n * Randomly shuffles elements in this array in-place using the specified [random] instance as the
source of randomness.\n */\n * See:
https://en.wikipedia.org/wiki/Fisher%20%93Yates\_shuffle#The\_modern\_algorithm\n
*/\n@SinceKotlin("1.4")\npublic fun ShortArray.shuffle(random: Random): Unit {\n    for (i in lastIndex downTo
1) {\n        val j = random.nextInt(i + 1)\n        val copy = this[i]\n        this[i] = this[j]\n        this[j] = copy\n
    }\n}\n\n/**\n * Randomly shuffles elements in this array in-place using the specified [random] instance as the
source of randomness.\n */\n * See:
https://en.wikipedia.org/wiki/Fisher%20%93Yates\_shuffle#The\_modern\_algorithm\n
*/\n@SinceKotlin("1.4")\npublic fun IntArray.shuffle(random: Random): Unit {\n    for (i in lastIndex downTo 1)
{\n        val j = random.nextInt(i + 1)\n        val copy = this[i]\n        this[i] = this[j]\n        this[j] = copy\n
    }\n}\n\n/**\n * Randomly shuffles elements in this array in-place using the specified [random] instance as the
source of randomness.\n */\n * See:
https://en.wikipedia.org/wiki/Fisher%20%93Yates\_shuffle#The\_modern\_algorithm\n
*/\n@SinceKotlin("1.4")\npublic fun LongArray.shuffle(random: Random): Unit {\n    for (i in lastIndex downTo
1) {\n        val j = random.nextInt(i + 1)\n        val copy = this[i]\n        this[i] = this[j]\n        this[j] = copy\n
    }\n}\n\n/**\n * Randomly shuffles elements in this array in-place using the specified [random] instance as the
source of randomness.\n */\n * See:
https://en.wikipedia.org/wiki/Fisher%20%93Yates\_shuffle#The\_modern\_algorithm\n
*/\n@SinceKotlin("1.4")\npublic fun FloatArray.shuffle(random: Random): Unit {\n    for (i in lastIndex downTo

```

```

1) {\n    val j = random.nextInt(i + 1)\n    val copy = this[i]\n    this[i] = this[j]\n    this[j] = copy\n}\n}\n\n/**\n * Randomly shuffles elements in this array in-place using the specified [random] instance as the\n source of randomness.\n * \n * See:\n https://en.wikipedia.org/wiki/Fisher%20%80%93Yates\_shuffle#The\_modern\_algorithm\n *\n @SinceKotlin("1.4")\n public fun DoubleArray.shuffle(random: Random): Unit {\n    for (i in lastIndex\n downTo 1) {\n        val j = random.nextInt(i + 1)\n        val copy = this[i]\n        this[i] = this[j]\n        this[j] = copy\n    }\n}\n\n/**\n * Randomly shuffles elements in this array in-place using the specified [random] instance as the\n source of randomness.\n * \n * See:\n https://en.wikipedia.org/wiki/Fisher%20%80%93Yates\_shuffle#The\_modern\_algorithm\n *\n @SinceKotlin("1.4")\n public fun BooleanArray.shuffle(random: Random): Unit {\n    for (i in lastIndex\n downTo 1) {\n        val j = random.nextInt(i + 1)\n        val copy = this[i]\n        this[i] = this[j]\n        this[j] = copy\n    }\n}\n\n/**\n * Randomly shuffles elements in this array in-place using the specified [random] instance as the\n source of randomness.\n * \n * See:\n https://en.wikipedia.org/wiki/Fisher%20%80%93Yates\_shuffle#The\_modern\_algorithm\n *\n @SinceKotlin("1.4")\n public fun CharArray.shuffle(random: Random): Unit {\n    for (i in lastIndex downTo\n 1) {\n        val j = random.nextInt(i + 1)\n        val copy = this[i]\n        this[i] = this[j]\n        this[j] = copy\n    }\n}\n\n/**\n * Sorts elements in the array in-place according to natural sort order of the value returned by specified\n [selector] function.\n * \n * The sort is _stable_. It means that equal elements preserve their order relative to each\n other after sorting.\n *\n public inline fun <T, R : Comparable<R>> Array<out T>.sortBy(crossinline selector: (T) -\n > R?): Unit {\n    if (size > 1) sortWith(compareBy(selector))\n}\n\n/**\n * Sorts elements in the array in-place\n descending according to natural sort order of the value returned by specified [selector] function.\n * \n * The sort is\n _stable_. It means that equal elements preserve their order relative to each other after sorting.\n *\n public inline fun\n <T, R : Comparable<R>> Array<out T>.sortByDescending(crossinline selector: (T) -> R?): Unit {\n    if (size > 1)\n sortWith(compareByDescending(selector))\n}\n\n/**\n * Sorts elements in the array in-place descending according\n to their natural sort order.\n * \n * The sort is _stable_. It means that equal elements preserve their order relative to\n each other after sorting.\n *\n public fun <T : Comparable<T>> Array<out T>.sortDescending(): Unit {\n    if (size > 1)\n sortWith(reverseOrder())\n}\n\n/**\n * Sorts elements in the array in-place descending according to their natural\n sort order.\n *\n public fun ByteArray.sortDescending(): Unit {\n    if (size > 1) {\n        sort()\n        reverse()\n    }\n}\n\n/**\n * Sorts elements in the array in-place descending according to their natural sort order.\n *\n public fun ShortArray.sortDescending(): Unit {\n    if (size > 1) {\n        sort()\n        reverse()\n    }\n}\n\n/**\n * Sorts\n elements in the array in-place descending according to their natural sort order.\n *\n public fun\n IntArray.sortDescending(): Unit {\n    if (size > 1) {\n        sort()\n        reverse()\n    }\n}\n\n/**\n * Sorts elements\n in the array in-place descending according to their natural sort order.\n *\n public fun\n LongArray.sortDescending():\n Unit {\n    if (size > 1) {\n        sort()\n        reverse()\n    }\n}\n\n/**\n * Sorts elements in the array in-place\n descending according to their natural sort order.\n *\n public fun\n FloatArray.sortDescending(): Unit {\n    if (size >\n 1) {\n        sort()\n        reverse()\n    }\n}\n\n/**\n * Sorts elements in the array in-place descending according to\n their natural sort order.\n *\n public fun\n DoubleArray.sortDescending(): Unit {\n    if (size > 1) {\n        sort()\n        reverse()\n    }\n}\n\n/**\n * Sorts elements in the array in-place descending according to their natural sort order.\n *\n public fun\n CharArray.sortDescending(): Unit {\n    if (size > 1) {\n        sort()\n        reverse()\n    }\n}\n\n/**\n * Returns a list of all elements sorted according to their natural sort order.\n * \n * The sort is _stable_. It means that\n equal elements preserve their order relative to each other after sorting.\n *\n public fun <T : Comparable<T>>\n Array<out T>.sorted(): List<T> {\n    return sortedArray().asList()\n}\n\n/**\n * Returns a list of all elements sorted\n according to their natural sort order.\n *\n public fun\n ByteArray.sorted(): List<Byte> {\n    return\n toTypedArray().apply { sort() }.asList()\n}\n\n/**\n * Returns a list of all elements sorted according to their natural\n sort order.\n *\n public fun\n ShortArray.sorted(): List<Short> {\n    return toTypedArray().apply { sort()\n }\n}.asList()\n}\n\n/**\n * Returns a list of all elements sorted according to their natural sort order.\n *\n public fun\n IntArray.sorted(): List<Int> {\n    return toTypedArray().apply { sort() }.asList()\n}\n\n/**\n * Returns a list of all\n elements sorted according to their natural sort order.\n *\n public fun\n LongArray.sorted(): List<Long> {\n    return

```

toTypedArray().apply { sort() }.asList()\n\n/\*\*\n \* Returns a list of all elements sorted according to their natural sort order.\n \*/\npublic fun FloatArray.sorted(): List<Float> {\n return toTypedArray().apply { sort() }\n}.asList()\n\n/\*\*\n \* Returns a list of all elements sorted according to their natural sort order.\n \*/\npublic fun DoubleArray.sorted(): List<Double> {\n return toTypedArray().apply { sort() }.asList()\n}\n\n/\*\*\n \* Returns a list of all elements sorted according to their natural sort order.\n \*/\npublic fun CharArray.sorted(): List<Char> {\n return toTypedArray().apply { sort() }.asList()\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted according to their natural sort order.\n \* \n \* The sort is `_stable_`. It means that equal elements preserve their order relative to each other after sorting.\n \*/\npublic fun <T : Comparable<T>> Array<T>.sortedArray(): Array<T> {\n if (isEmpty()) return this\n return this.copyOf().apply { sort() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted according to their natural sort order.\n \*/\npublic fun ByteArray.sortedArray(): ByteArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sort() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted according to their natural sort order.\n \*/\npublic fun ShortArray.sortedArray(): ShortArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sort() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted according to their natural sort order.\n \*/\npublic fun IntArray.sortedArray(): IntArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sort() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted according to their natural sort order.\n \*/\npublic fun LongArray.sortedArray(): LongArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sort() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted according to their natural sort order.\n \*/\npublic fun FloatArray.sortedArray(): FloatArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sort() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted according to their natural sort order.\n \*/\npublic fun DoubleArray.sortedArray(): DoubleArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sort() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted according to their natural sort order.\n \*/\npublic fun CharArray.sortedArray(): CharArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sort() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted descending according to their natural sort order.\n \* \n \* The sort is `_stable_`. It means that equal elements preserve their order relative to each other after sorting.\n \*/\npublic fun <T : Comparable<T>> Array<T>.sortedArrayDescending(): Array<T> {\n if (isEmpty()) return this\n return this.copyOf().apply { sortWith(reverseOrder()) }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted descending according to their natural sort order.\n \*/\npublic fun ByteArray.sortedArrayDescending(): ByteArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sortDescending() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted descending according to their natural sort order.\n \*/\npublic fun ShortArray.sortedArrayDescending(): ShortArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sortDescending() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted descending according to their natural sort order.\n \*/\npublic fun IntArray.sortedArrayDescending(): IntArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sortDescending() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted descending according to their natural sort order.\n \*/\npublic fun LongArray.sortedArrayDescending(): LongArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sortDescending() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted descending according to their natural sort order.\n \*/\npublic fun FloatArray.sortedArrayDescending(): FloatArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sortDescending() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted descending according to their natural sort order.\n \*/\npublic fun DoubleArray.sortedArrayDescending(): DoubleArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sortDescending() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted descending according to their natural sort order.\n \*/\npublic fun CharArray.sortedArrayDescending(): CharArray {\n if (isEmpty()) return this\n return this.copyOf().apply { sortDescending() }\n}\n\n/\*\*\n \* Returns an array with all elements of this array sorted according the specified [comparator].\n \* \n \* The sort is `_stable_`. It means that equal elements preserve their order relative to each other after sorting.\n \*/\npublic fun <T> Array<out T>.sortedArrayWith(comparator: Comparator<in T>): Array<out T> {\n if (isEmpty()) return this\n return this.copyOf().apply { sortWith(comparator) }\n}\n\n/\*\*\n \* Returns a list of all elements sorted according to natural sort order of the value returned by specified [selector] function.\n \* \n \*

The sort is `_stable_`. It means that equal elements preserve their order relative to each other after sorting.

```

@sample samples.collections.Collections.Sorting.sortedBy
Array<out T>.sortedBy(crossinline selector: (T) -> R?): List<T> {
    return
    sortedWith(compareBy(selector))
}

```

Returns a list of all elements sorted according to natural sort order of the value returned by specified [selector] function.

```

@sample
samples.collections.Collections.Sorting.sortedBy
ByteArrayList.sortedBy(crossinline selector: (Byte) -> R?): List<Byte> {
    return
    sortedWith(compareBy(selector))
}

```

Returns a list of all elements sorted according to natural sort order of the value returned by specified [selector] function.

```

@sample
samples.collections.Collections.Sorting.sortedBy
ShortArrayList.sortedBy(crossinline selector: (Short) -> R?): List<Short> {
    return
    sortedWith(compareBy(selector))
}

```

Returns a list of all elements sorted according to natural sort order of the value returned by specified [selector] function.

```

@sample
samples.collections.Collections.Sorting.sortedBy
IntArray.sortedBy(crossinline selector: (Int) -> R?): List<Int> {
    return
    sortedWith(compareBy(selector))
}

```

Returns a list of all elements sorted according to natural sort order of the value returned by specified [selector] function.

```

@sample
samples.collections.Collections.Sorting.sortedBy
LongArrayList.sortedBy(crossinline selector: (Long) -> R?): List<Long> {
    return
    sortedWith(compareBy(selector))
}

```

Returns a list of all elements sorted according to natural sort order of the value returned by specified [selector] function.

```

@sample
samples.collections.Collections.Sorting.sortedBy
FloatArrayList.sortedBy(crossinline selector: (Float) -> R?): List<Float> {
    return
    sortedWith(compareBy(selector))
}

```

Returns a list of all elements sorted according to natural sort order of the value returned by specified [selector] function.

```

@sample
samples.collections.Collections.Sorting.sortedBy
DoubleArrayList.sortedBy(crossinline selector: (Double) -> R?): List<Double> {
    return
    sortedWith(compareBy(selector))
}

```

Returns a list of all elements sorted according to natural sort order of the value returned by specified [selector] function.

```

@sample
samples.collections.Collections.Sorting.sortedBy
BooleanArrayList.sortedBy(crossinline selector: (Boolean) -> R?): List<Boolean> {
    return
    sortedWith(compareBy(selector))
}

```

Returns a list of all elements sorted according to natural sort order of the value returned by specified [selector] function.

```

@sample
samples.collections.Collections.Sorting.sortedBy
CharArray.sortedBy(crossinline selector: (Char) -> R?): List<Char> {
    return
    sortedWith(compareBy(selector))
}

```

Returns a list of all elements sorted descending according to natural sort order of the value returned by specified [selector] function. The sort is `_stable_`. It means that equal elements preserve their order relative to each other after sorting.

```

@sample
samples.collections.Collections.Sorting.sortedByDescending
Array<out T>.sortedByDescending(crossinline selector: (T) -> R?): List<T> {
    return
    sortedWith(compareByDescending(selector))
}

```

Returns a list of all elements sorted descending according to natural sort order of the value returned by specified [selector] function.

```

@sample
samples.collections.Collections.Sorting.sortedByDescending
ByteArrayList.sortedByDescending(crossinline selector: (Byte) -> R?): List<Byte> {
    return
    sortedWith(compareByDescending(selector))
}

```

Returns a list of all elements sorted descending according to natural sort order of the value returned by specified [selector] function.

```

@sample
samples.collections.Collections.Sorting.sortedByDescending
ShortArrayList.sortedByDescending(crossinline selector: (Short) -> R?): List<Short> {
    return
    sortedWith(compareByDescending(selector))
}

```

Returns a list of all elements sorted descending according to natural sort order of the value returned by specified [selector] function.

```

@sample
samples.collections.Collections.Sorting.sortedByDescending
IntArray.sortedByDescending(crossinline selector: (Int) -> R?): List<Int> {
    return
    sortedWith(compareByDescending(selector))
}

```

Returns a list of all elements sorted descending according to natural sort order of the value returned by specified [selector] function.

sortedWith(compareByDescending(selector))\n\n/\*\*\n \* Returns a list of all elements sorted descending according to natural sort order of the value returned by specified [selector] function.\n \*/\npublic inline fun <R : Comparable<R>> LongArray.sortedByDescending(crossinline selector: (Long) -> R?): List<Long> {\n return sortedWith(compareByDescending(selector))\n}\n\n/\*\*\n \* Returns a list of all elements sorted descending according to natural sort order of the value returned by specified [selector] function.\n \*/\npublic inline fun <R : Comparable<R>> FloatArray.sortedByDescending(crossinline selector: (Float) -> R?): List<Float> {\n return sortedWith(compareByDescending(selector))\n}\n\n/\*\*\n \* Returns a list of all elements sorted descending according to natural sort order of the value returned by specified [selector] function.\n \*/\npublic inline fun <R : Comparable<R>> DoubleArray.sortedByDescending(crossinline selector: (Double) -> R?): List<Double> {\n return sortedWith(compareByDescending(selector))\n}\n\n/\*\*\n \* Returns a list of all elements sorted descending according to natural sort order of the value returned by specified [selector] function.\n \*/\npublic inline fun <R : Comparable<R>> BooleanArray.sortedByDescending(crossinline selector: (Boolean) -> R?): List<Boolean> {\n return sortedWith(compareByDescending(selector))\n}\n\n/\*\*\n \* Returns a list of all elements sorted descending according to natural sort order of the value returned by specified [selector] function.\n \*/\npublic inline fun <R : Comparable<R>> CharArray.sortedByDescending(crossinline selector: (Char) -> R?): List<Char> {\n return sortedWith(compareByDescending(selector))\n}\n\n/\*\*\n \* Returns a list of all elements sorted descending according to their natural sort order.\n \* \n \* The sort is `_stable_`. It means that equal elements preserve their order relative to each other after sorting.\n \*/\npublic fun <T : Comparable<T>> Array<out T>.sortedDescending(): List<T> {\n return sortedWith(reverseOrder())\n}\n\n/\*\*\n \* Returns a list of all elements sorted descending according to their natural sort order.\n \*/\npublic fun ByteArray.sortedDescending(): List<Byte> {\n return copyOf().apply { sort() }.reversed()\n}\n\n/\*\*\n \* Returns a list of all elements sorted descending according to their natural sort order.\n \*/\npublic fun ShortArray.sortedDescending(): List<Short> {\n return copyOf().apply { sort() }.reversed()\n}\n\n/\*\*\n \* Returns a list of all elements sorted descending according to their natural sort order.\n \*/\npublic fun IntArray.sortedDescending(): List<Int> {\n return copyOf().apply { sort() }.reversed()\n}\n\n/\*\*\n \* Returns a list of all elements sorted descending according to their natural sort order.\n \*/\npublic fun LongArray.sortedDescending(): List<Long> {\n return copyOf().apply { sort() }.reversed()\n}\n\n/\*\*\n \* Returns a list of all elements sorted descending according to their natural sort order.\n \*/\npublic fun FloatArray.sortedDescending(): List<Float> {\n return copyOf().apply { sort() }.reversed()\n}\n\n/\*\*\n \* Returns a list of all elements sorted descending according to their natural sort order.\n \*/\npublic fun DoubleArray.sortedDescending(): List<Double> {\n return copyOf().apply { sort() }.reversed()\n}\n\n/\*\*\n \* Returns a list of all elements sorted descending according to their natural sort order.\n \*/\npublic fun CharArray.sortedDescending(): List<Char> {\n return copyOf().apply { sort() }.reversed()\n}\n\n/\*\*\n \* Returns a list of all elements sorted according to the specified [comparator].\n \* \n \* The sort is `_stable_`. It means that equal elements preserve their order relative to each other after sorting.\n \*/\npublic fun <T> Array<out T>.sortedWith(comparator: Comparator<in T>): List<T> {\n return sortedArrayWith(comparator).asList()\n}\n\n/\*\*\n \* Returns a list of all elements sorted according to the specified [comparator].\n \*/\npublic fun ByteArray.sortedWith(comparator: Comparator<in Byte>): List<Byte> {\n return toTypedArray().apply { sortWith(comparator) }.asList()\n}\n\n/\*\*\n \* Returns a list of all elements sorted according to the specified [comparator].\n \*/\npublic fun ShortArray.sortedWith(comparator: Comparator<in Short>): List<Short> {\n return toTypedArray().apply { sortWith(comparator) }.asList()\n}\n\n/\*\*\n \* Returns a list of all elements sorted according to the specified [comparator].\n \*/\npublic fun IntArray.sortedWith(comparator: Comparator<in Int>): List<Int> {\n return toTypedArray().apply { sortWith(comparator) }.asList()\n}\n\n/\*\*\n \* Returns a list of all elements sorted according to the specified [comparator].\n \*/\npublic fun LongArray.sortedWith(comparator: Comparator<in Long>): List<Long> {\n return toTypedArray().apply { sortWith(comparator) }.asList()\n}\n\n/\*\*\n \* Returns a list of all elements sorted according to the specified [comparator].\n \*/\npublic fun FloatArray.sortedWith(comparator: Comparator<in Float>): List<Float> {\n return toTypedArray().apply { sortWith(comparator) }.asList()\n}\n\n/\*\*\n \* Returns a list of all elements sorted according to the specified [comparator].\n \*/\npublic fun DoubleArray.sortedWith(comparator: Comparator<in Double>):

`List<Double> { \n return toTypedArray().apply { sortWith(comparator) }.asList()\n }\n\n/**\n * Returns a list of all elements sorted according to the specified [comparator].\n * \n\npublic fun BooleanArray.sortedWith(comparator: Comparator<in Boolean>): List<Boolean> { \n return toTypedArray().apply { sortWith(comparator) }.asList()\n }\n\n/**\n * Returns a list of all elements sorted according to the specified [comparator].\n * \n\npublic fun CharArray.sortedWith(comparator: Comparator<in Char>): List<Char> { \n return toTypedArray().apply { sortWith(comparator) }.asList()\n }\n\n/**\n * Returns a [List] that wraps the original array.\n * \n\npublic expect fun <T> Array<out T>.asList(): List<T>\n\n/**\n * Returns a [List] that wraps the original array.\n * \n\npublic expect fun ByteArray.asList(): List<Byte>\n\n/**\n * Returns a [List] that wraps the original array.\n * \n\npublic expect fun ShortArray.asList(): List<Short>\n\n/**\n * Returns a [List] that wraps the original array.\n * \n\npublic expect fun IntArray.asList(): List<Int>\n\n/**\n * Returns a [List] that wraps the original array.\n * \n\npublic expect fun LongArray.asList(): List<Long>\n\n/**\n * Returns a [List] that wraps the original array.\n * \n\npublic expect fun FloatArray.asList(): List<Float>\n\n/**\n * Returns a [List] that wraps the original array.\n * \n\npublic expect fun DoubleArray.asList(): List<Double>\n\n/**\n * Returns a [List] that wraps the original array.\n * \n\npublic expect fun BooleanArray.asList(): List<Boolean>\n\n/**\n * Returns a [List] that wraps the original array.\n * \n\npublic expect fun CharArray.asList(): List<Char>\n\n/**\n * Returns `true` if the two specified arrays are deeply equal to one another,\n * i.e. contain the same number of the same elements in the same order.\n * \n\n * If two corresponding elements are nested arrays, they are also compared deeply.\n * If any of arrays contains itself on any nesting level the behavior is undefined.\n * \n\n * The elements of other types are compared for equality with the [equals][Any.equals] function.\n * For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n * \n\n@SinceKotlin("1.1")\n@kotlin.internal.LowPriorityInOverloadResolution\n\npublic expect infix fun <T> Array<out T>.contentDeepEquals(other: Array<out T>): Boolean\n\n/**\n * Returns `true` if the two specified arrays are deeply equal to one another,\n * i.e. contain the same number of the same elements in the same order.\n * \n\n * The specified arrays are also considered deeply equal if both are `null`.\n * \n\n * If two corresponding elements are nested arrays, they are also compared deeply.\n * If any of arrays contains itself on any nesting level the behavior is undefined.\n * \n\n * The elements of other types are compared for equality with the [equals][Any.equals] function.\n * For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n * \n\n@SinceKotlin("1.4")\n\npublic expect infix fun <T> Array<out T>?.contentDeepEquals(other: Array<out T>?): Boolean\n\n/**\n * Returns a hash code based on the contents of this array as if it is [List].\n * \n\n * Nested arrays are treated as lists too.\n * \n\n * If any of arrays contains itself on any nesting level the behavior is undefined.\n * \n\n@SinceKotlin("1.1")\n@kotlin.internal.LowPriorityInOverloadResolution\n\npublic expect fun <T> Array<out T>.contentDeepHashCode(): Int\n\n/**\n * Returns a hash code based on the contents of this array as if it is [List].\n * \n\n * Nested arrays are treated as lists too.\n * \n\n * If any of arrays contains itself on any nesting level the behavior is undefined.\n * \n\n@SinceKotlin("1.4")\n\npublic expect fun <T> Array<out T>?.contentDeepHashCode(): Int\n\n/**\n * Returns a string representation of the contents of this array as if it is a [List].\n * \n\n * Nested arrays are treated as lists too.\n * \n\n * If any of arrays contains itself on any nesting level that reference\n * is rendered as `\"[...]\"` to prevent recursion.\n * \n\n * @sample samples.collections.Arrays.ContentOperations.contentDeepToString\n * \n\n@SinceKotlin("1.1")\n@kotlin.internal.LowPriorityInOverloadResolution\n\npublic expect fun <T> Array<out T>.contentDeepToString(): String\n\n/**\n * Returns a string representation of the contents of this array as if it is a [List].\n * \n\n * Nested arrays are treated as lists too.\n * \n\n * If any of arrays contains itself on any nesting level that reference\n * is rendered as `\"[...]\"` to prevent recursion.\n * \n\n * @sample samples.collections.Arrays.ContentOperations.contentDeepToString\n * \n\n@SinceKotlin("1.4")\n\npublic expect fun <T> Array<out T>?.contentDeepToString(): String\n\n/**\n * Returns `true` if the two specified arrays are structurally equal to one another,\n * i.e. contain the same number of the same elements in the same order.\n * \n\n * The elements are compared for equality with the [equals][Any.equals] function.\n * For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n * \n\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\n\npublic expect infix fun <T> Array<out T>.contentEquals(other: Array<out T>): Boolean\n\n/**\n * Returns `true` if the`

two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.

`@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")@SinceKotlin("1.1")@DeprecatedSinceKotlin(hiddenSince = "1.4")`public expect infix fun ByteArray.contentEquals(other: ByteArray): Boolean

Returns true if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.

`@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")@SinceKotlin("1.1")@DeprecatedSinceKotlin(hiddenSince = "1.4")`public expect infix fun ShortArray.contentEquals(other: ShortArray): Boolean

Returns true if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.

`@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")@SinceKotlin("1.1")@DeprecatedSinceKotlin(hiddenSince = "1.4")`public expect infix fun IntArray.contentEquals(other: IntArray): Boolean

Returns true if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.

`@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")@SinceKotlin("1.1")@DeprecatedSinceKotlin(hiddenSince = "1.4")`public expect infix fun LongArray.contentEquals(other: LongArray): Boolean

Returns true if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.

`@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")@SinceKotlin("1.1")@DeprecatedSinceKotlin(hiddenSince = "1.4")`public expect infix fun FloatArray.contentEquals(other: FloatArray): Boolean

Returns true if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.

`@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")@SinceKotlin("1.1")@DeprecatedSinceKotlin(hiddenSince = "1.4")`public expect infix fun DoubleArray.contentEquals(other: DoubleArray): Boolean

Returns true if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.

`@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")@SinceKotlin("1.1")@DeprecatedSinceKotlin(hiddenSince = "1.4")`public expect infix fun BooleanArray.contentEquals(other: BooleanArray): Boolean

Returns true if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.

`@SinceKotlin("1.4")`public expect infix fun <T> Array<out T>?.contentEquals(other: Array<out T>?): Boolean

Returns true if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements

in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.

```

@SinceKotlin("1.4")
public expect infix fun ByteArray?.contentEquals(other: ByteArray?): Boolean
Returns true if the two specified arrays are structurally equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.
@SinceKotlin("1.4")
public expect infix fun ShortArray?.contentEquals(other: ShortArray?): Boolean
Returns true if the two specified arrays are structurally equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.
@SinceKotlin("1.4")
public expect infix fun IntArray?.contentEquals(other: IntArray?): Boolean
Returns true if the two specified arrays are structurally equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.
@SinceKotlin("1.4")
public expect infix fun LongArray?.contentEquals(other: LongArray?): Boolean
Returns true if the two specified arrays are structurally equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.
@SinceKotlin("1.4")
public expect infix fun FloatArray?.contentEquals(other: FloatArray?): Boolean
Returns true if the two specified arrays are structurally equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.
@SinceKotlin("1.4")
public expect infix fun DoubleArray?.contentEquals(other: DoubleArray?): Boolean
Returns true if the two specified arrays are structurally equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.
@SinceKotlin("1.4")
public expect infix fun BooleanArray?.contentEquals(other: BooleanArray?): Boolean
Returns true if the two specified arrays are structurally equal to one another, i.e. contain the same number of the same elements in the same order. The elements are compared for equality with the [equals][Any.equals] function. For floating point numbers it means that NaN is equal to itself and -0.0 is not equal to 0.0.
@SinceKotlin("1.4")
public expect infix fun CharArray?.contentEquals(other: CharArray?): Boolean
Returns a hash code based on the contents of this array as if it is [List].
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")
@SinceKotlin("1.1")
@DeprecatedSinceKotlin(hiddenSince = "1.4")
public expect fun <T> Array<out T>.contentHashCode(): Int
Returns a hash code based on the contents of this array as if it is [List].
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")
@SinceKotlin("1.1")
@DeprecatedSinceKotlin(hiddenSince = "1.4")
public expect fun ByteArray.contentHashCode(): Int
Returns a hash code based on the contents of this array as if it is [List].
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")
@SinceKotlin("1.1")
@DeprecatedSinceKotlin(hiddenSince = "1.4")
public expect fun ShortArray.contentHashCode(): Int
Returns a hash code based on the contents of this array as if it is [List].
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")
@SinceKotlin("1.1")
@DeprecatedSinceKotlin(hiddenSince = "1.4")
public expect fun IntArray.contentHashCode(): Int
Returns a hash code based on the contents of this array as if it is [List].
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")
@SinceKotlin("1.1")
@DeprecatedSinceKotlin(hiddenSince = "1.4")
public expect fun LongArray.contentHashCode(): Int
Returns a hash code based on the contents of this array as if it is [List].
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")

```

warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic expect fun  
FloatArray.contentHashCode(): Int\n\n/\*\*\n \* Returns a hash code based on the contents of this array as if it is  
[List].\n \* \n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation  
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic expect fun  
DoubleArray.contentHashCode(): Int\n\n/\*\*\n \* Returns a hash code based on the contents of this array as if it is  
[List].\n \* \n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation  
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic expect fun  
BooleanArray.contentHashCode(): Int\n\n/\*\*\n \* Returns a hash code based on the contents of this array as if it is  
[List].\n \* \n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation  
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic expect fun  
CharArray.contentHashCode(): Int\n\n/\*\*\n \* Returns a hash code based on the contents of this array as if it is  
[List].\n \* \n@SinceKotlin(\"1.4\")\npublic expect fun <T> Array<out T>?.contentHashCode(): Int\n\n/\*\*\n \* Returns a hash code based on the contents of this array as if it is [List].\n \* \n@SinceKotlin(\"1.4\")\npublic expect fun  
ByteArray?.contentHashCode(): Int\n\n/\*\*\n \* Returns a hash code based on the contents of this array as if it is  
[List].\n \* \n@SinceKotlin(\"1.4\")\npublic expect fun ShortArray?.contentHashCode(): Int\n\n/\*\*\n \* Returns a  
hash code based on the contents of this array as if it is [List].\n \* \n@SinceKotlin(\"1.4\")\npublic expect fun  
IntArray?.contentHashCode(): Int\n\n/\*\*\n \* Returns a hash code based on the contents of this array as if it is  
[List].\n \* \n@SinceKotlin(\"1.4\")\npublic expect fun LongArray?.contentHashCode(): Int\n\n/\*\*\n \* Returns a  
hash code based on the contents of this array as if it is [List].\n \* \n@SinceKotlin(\"1.4\")\npublic expect fun  
FloatArray?.contentHashCode(): Int\n\n/\*\*\n \* Returns a hash code based on the contents of this array as if it is  
[List].\n \* \n@SinceKotlin(\"1.4\")\npublic expect fun DoubleArray?.contentHashCode(): Int\n\n/\*\*\n \* Returns a  
hash code based on the contents of this array as if it is [List].\n \* \n@SinceKotlin(\"1.4\")\npublic expect fun  
BooleanArray?.contentHashCode(): Int\n\n/\*\*\n \* Returns a hash code based on the contents of this array as if it is  
[List].\n \* \n@SinceKotlin(\"1.4\")\npublic expect fun CharArray?.contentHashCode(): Int\n\n/\*\*\n \* Returns a  
string representation of the contents of the specified array as if it is [List].\n \* \n \* \n \* @sample  
samples.collections.Arrays.ContentOperations.contentToString\n \* \n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation  
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic  
expect fun <T> Array<out T>.contentToString(): String\n\n/\*\*\n \* Returns a string representation of the contents of  
the specified array as if it is [List].\n \* \n \* \n \* @sample  
samples.collections.Arrays.ContentOperations.contentToString\n \* \n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation  
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic  
expect fun ByteArray.contentToString(): String\n\n/\*\*\n \* Returns a string representation of the contents of the  
specified array as if it is [List].\n \* \n \* \n \* @sample samples.collections.Arrays.ContentOperations.contentToString\n \* \n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation  
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic expect fun  
ShortArray.contentToString(): String\n\n/\*\*\n \* Returns a string representation of the contents of the specified array  
as if it is [List].\n \* \n \* \n \* @sample samples.collections.Arrays.ContentOperations.contentToString\n \* \n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation  
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic expect fun  
IntArray.contentToString(): String\n\n/\*\*\n \* Returns a string representation of the contents of the specified array as  
if it is [List].\n \* \n \* \n \* @sample samples.collections.Arrays.ContentOperations.contentToString\n \* \n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation  
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic expect fun  
LongArray.contentToString(): String\n\n/\*\*\n \* Returns a string representation of the contents of the specified array  
as if it is [List].\n \* \n \* \n \* @sample samples.collections.Arrays.ContentOperations.contentToString\n \* \n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation  
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic expect fun  
FloatArray.contentToString(): String\n\n/\*\*\n \* Returns a string representation of the contents of the specified array

as if it is [List].\n \* \n \* @sample samples.collections.Arrays.ContentOperations.contentToString\n

```
*\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic expect fun
DoubleArray.contentToString(): String\n\n/**\n * Returns a string representation of the contents of the specified
array as if it is [List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n
*\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic expect fun
BooleanArray.contentToString(): String\n\n/**\n * Returns a string representation of the contents of the specified
array as if it is [List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n
*\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic expect fun
CharArray.contentToString(): String\n\n/**\n * Returns a string representation of the contents of the specified array
as if it is [List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n
*\n@SinceKotlin("1.4")\npublic expect fun <T> Array<out T>?.contentToString(): String\n\n/**\n * Returns a
string representation of the contents of the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n *\n@SinceKotlin("1.4")\npublic expect fun
ByteArray?.contentToString(): String\n\n/**\n * Returns a string representation of the contents of the specified array
as if it is [List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n
*\n@SinceKotlin("1.4")\npublic expect fun ShortArray?.contentToString(): String\n\n/**\n * Returns a string
representation of the contents of the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n *\n@SinceKotlin("1.4")\npublic expect fun
IntArray?.contentToString(): String\n\n/**\n * Returns a string representation of the contents of the specified array
as if it is [List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n
*\n@SinceKotlin("1.4")\npublic expect fun LongArray?.contentToString(): String\n\n/**\n * Returns a string
representation of the contents of the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n *\n@SinceKotlin("1.4")\npublic expect fun
FloatArray?.contentToString(): String\n\n/**\n * Returns a string representation of the contents of the specified
array as if it is [List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n
*\n@SinceKotlin("1.4")\npublic expect fun DoubleArray?.contentToString(): String\n\n/**\n * Returns a string
representation of the contents of the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n *\n@SinceKotlin("1.4")\npublic expect fun
BooleanArray?.contentToString(): String\n\n/**\n * Returns a string representation of the contents of the specified
array as if it is [List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n
*\n@SinceKotlin("1.4")\npublic expect fun CharArray?.contentToString(): String\n\n/**\n * Copies this array or
its subrange into the [destination] array and returns that array.\n * \n * It's allowed to pass the same array in the
[destination] and even specify the subrange so that it overlaps with the destination range.\n * \n * @param
destination the array to copy to.\n * @param destinationOffset the position in the [destination] array to copy to, 0 by
default.\n * @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default.\n * @param
endIndex the end (exclusive) of the subrange to copy, size of this array by default.\n * \n * @throws
IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this
array indices or when `startIndex > endIndex`.\n * @throws IndexOutOfBoundsException when the subrange
doesn't fit into the [destination] array starting at the specified [destinationOffset],\n * or when that index is out of the
[destination] array indices range.\n * \n * @return the [destination] array.\n *\n@SinceKotlin("1.3")\npublic
expect fun <T> Array<out T>.copyInto(destination: Array<T>, destinationOffset: Int = 0, startIndex: Int = 0,
endIndex: Int = size): Array<T>\n\n/**\n * Copies this array or its subrange into the [destination] array and returns
that array.\n * \n * It's allowed to pass the same array in the [destination] and even specify the subrange so that it
overlaps with the destination range.\n * \n * @param destination the array to copy to.\n * @param destinationOffset
the position in the [destination] array to copy to, 0 by default.\n * @param startIndex the beginning (inclusive) of
```

the subrange to copy, 0 by default. \n \* @param endIndex the end (exclusive) of the subrange to copy, size of this array by default. \n \* \n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`. \n \* @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset], \n \* or when that index is out of the [destination] array indices range. \n \* \n \* @return the [destination] array. \n \*/\n\n@SinceKotlin("1.3")\npublic expect fun ByteArray.copyInto(destination: ByteArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size): ByteArray\n\n/\*\*\n \* Copies this array or its subrange into the [destination] array and returns that array. \n \* \n \* It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range. \n \* \n \* @param destination the array to copy to. \n \* @param destinationOffset the position in the [destination] array to copy to, 0 by default. \n \* @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default. \n \* @param endIndex the end (exclusive) of the subrange to copy, size of this array by default. \n \* \n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`. \n \* @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset], \n \* or when that index is out of the [destination] array indices range. \n \* \n \* @return the [destination] array. \n \*/\n\n@SinceKotlin("1.3")\npublic expect fun ShortArray.copyInto(destination: ShortArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size): ShortArray\n\n/\*\*\n \* Copies this array or its subrange into the [destination] array and returns that array. \n \* \n \* It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range. \n \* \n \* @param destination the array to copy to. \n \* @param destinationOffset the position in the [destination] array to copy to, 0 by default. \n \* @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default. \n \* @param endIndex the end (exclusive) of the subrange to copy, size of this array by default. \n \* \n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`. \n \* @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset], \n \* or when that index is out of the [destination] array indices range. \n \* \n \* @return the [destination] array. \n \*/\n\n@SinceKotlin("1.3")\npublic expect fun IntArray.copyInto(destination: IntArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size): IntArray\n\n/\*\*\n \* Copies this array or its subrange into the [destination] array and returns that array. \n \* \n \* It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range. \n \* \n \* @param destination the array to copy to. \n \* @param destinationOffset the position in the [destination] array to copy to, 0 by default. \n \* @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default. \n \* @param endIndex the end (exclusive) of the subrange to copy, size of this array by default. \n \* \n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`. \n \* @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset], \n \* or when that index is out of the [destination] array indices range. \n \* \n \* @return the [destination] array. \n \*/\n\n@SinceKotlin("1.3")\npublic expect fun LongArray.copyInto(destination: LongArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size): LongArray\n\n/\*\*\n \* Copies this array or its subrange into the [destination] array and returns that array. \n \* \n \* It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range. \n \* \n \* @param destination the array to copy to. \n \* @param destinationOffset the position in the [destination] array to copy to, 0 by default. \n \* @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default. \n \* @param endIndex the end (exclusive) of the subrange to copy, size of this array by default. \n \* \n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`. \n \* @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset], \n \* or when that index is out of the [destination] array indices range. \n \* \n \* @return the [destination] array. \n \*/\n\n@SinceKotlin("1.3")\npublic expect fun FloatArray.copyInto(destination: FloatArray, destinationOffset: Int =

0, startIndex: Int = 0, endIndex: Int = size): FloatArray\n\n/\*\*\n \* Copies this array or its subrange into the [destination] array and returns that array.\n \* \n \* It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range.\n \* \n \* @param destination the array to copy to.\n \* @param destinationOffset the position in the [destination] array to copy to, 0 by default.\n \* @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default.\n \* @param endIndex the end (exclusive) of the subrange to copy, size of this array by default.\n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`.\n \* @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],\n \* or when that index is out of the [destination] array indices range.\n \* @return the [destination] array.\n \*/\n\n@SinceKotlin("1.3")\npublic expect fun

DoubleArray.copyInto(destination: DoubleArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size): DoubleArray\n\n/\*\*\n \* Copies this array or its subrange into the [destination] array and returns that array.\n \* \n \* It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range.\n \* \n \* @param destination the array to copy to.\n \* @param destinationOffset the position in the [destination] array to copy to, 0 by default.\n \* @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default.\n \* @param endIndex the end (exclusive) of the subrange to copy, size of this array by default.\n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`.\n \* @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],\n \* or when that index is out of the [destination] array indices range.\n \* @return the [destination] array.\n

\*/\n\n@SinceKotlin("1.3")\npublic expect fun BooleanArray.copyInto(destination: BooleanArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size): BooleanArray\n\n/\*\*\n \* Copies this array or its subrange into the [destination] array and returns that array.\n \* \n \* It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range.\n \* \n \* @param destination the array to copy to.\n \* @param destinationOffset the position in the [destination] array to copy to, 0 by default.\n \* @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default.\n \* @param endIndex the end (exclusive) of the subrange to copy, size of this array by default.\n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`.\n \* @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],\n \* or when that index is out of the [destination] array indices range.\n \* @return the [destination] array.\n \*/\n\n@SinceKotlin("1.3")\npublic expect fun

CharArray.copyInto(destination: CharArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size):

CharArray\n\n/\*\*\n \* Returns new array which is a copy of the original array.\n \* \n \* @sample samples.collections.Arrays.CopyOfOperations.copyOfOf\n

\*/\n\n@Suppress("NO\_ACTUAL\_FOR\_EXPECT")\npublic expect fun <T> Array<T>.copyOf():

Array<T>\n\n/\*\*\n \* Returns new array which is a copy of the original array.\n \* \n \* @sample samples.collections.Arrays.CopyOfOperations.copyOfOf\n \*/\n\npublic expect fun ByteArray.copyOf():

ByteArray\n\n/\*\*\n \* Returns new array which is a copy of the original array.\n \* \n \* @sample samples.collections.Arrays.CopyOfOperations.copyOfOf\n \*/\n\npublic expect fun ShortArray.copyOf():

ShortArray\n\n/\*\*\n \* Returns new array which is a copy of the original array.\n \* \n \* @sample

samples.collections.Arrays.CopyOfOperations.copyOfOf\n \*/\n\npublic expect fun IntArray.copyOf(): IntArray\n\n/\*\*\n \* Returns new array which is a copy of the original array.\n \* \n \* @sample

samples.collections.Arrays.CopyOfOperations.copyOfOf\n \*/\n\npublic expect fun LongArray.copyOf():

LongArray\n\n/\*\*\n \* Returns new array which is a copy of the original array.\n \* \n \* @sample

samples.collections.Arrays.CopyOfOperations.copyOfOf\n \*/\n\npublic expect fun FloatArray.copyOf():

FloatArray\n\n/\*\*\n \* Returns new array which is a copy of the original array.\n \* \n \* @sample

samples.collections.Arrays.CopyOfOperations.copyOfOf\n \*/\n\npublic expect fun DoubleArray.copyOf():

DoubleArray\n\n/\*\*\n \* Returns new array which is a copy of the original array.\n \* \n \* @sample

samples.collections.Arrays.CopyOfOperations.copyOfOf\n \*^/npublic expect fun BooleanArray.copyOfOf():  
 BooleanArray\n/n/\*\*\n \* Returns new array which is a copy of the original array.\n \* \n \* @sample  
 samples.collections.Arrays.CopyOfOperations.copyOfOf\n \*^/npublic expect fun CharArray.copyOfOf():  
 CharArray\n/n/\*\*\n \* Returns new array which is a copy of the original array, resized to the given [newSize].\n \*  
 The copy is either truncated or padded at the end with zero values if necessary.\n \* \n \* - If [newSize] is less than the  
 size of the original array, the copy array is truncated to the [newSize].\n \* - If [newSize] is greater than the size of  
 the original array, the extra elements in the copy array are filled with zero values.\n \* \n \* @sample  
 samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n \*^/npublic expect fun  
 ByteArray.copyOfOf(newSize: Int): ByteArray\n/n/\*\*\n \* Returns new array which is a copy of the original array,  
 resized to the given [newSize].\n \* The copy is either truncated or padded at the end with zero values if necessary.\n  
 \* \n \* - If [newSize] is less than the size of the original array, the copy array is truncated to the [newSize].\n \* - If  
 [newSize] is greater than the size of the original array, the extra elements in the copy array are filled with zero  
 values.\n \* \n \* @sample samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n \*^/npublic  
 expect fun ShortArray.copyOfOf(newSize: Int): ShortArray\n/n/\*\*\n \* Returns new array which is a copy of the  
 original array, resized to the given [newSize].\n \* The copy is either truncated or padded at the end with zero values  
 if necessary.\n \* \n \* - If [newSize] is less than the size of the original array, the copy array is truncated to the  
 [newSize].\n \* - If [newSize] is greater than the size of the original array, the extra elements in the copy array are  
 filled with zero values.\n \* \n \* @sample samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n  
 \*^/npublic expect fun IntArray.copyOfOf(newSize: Int): IntArray\n/n/\*\*\n \* Returns new array which is a copy of the  
 original array, resized to the given [newSize].\n \* The copy is either truncated or padded at the end with zero values  
 if necessary.\n \* \n \* - If [newSize] is less than the size of the original array, the copy array is truncated to the  
 [newSize].\n \* - If [newSize] is greater than the size of the original array, the extra elements in the copy array are  
 filled with zero values.\n \* \n \* @sample samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n  
 \*^/npublic expect fun LongArray.copyOfOf(newSize: Int): LongArray\n/n/\*\*\n \* Returns new array which is a copy of  
 the original array, resized to the given [newSize].\n \* The copy is either truncated or padded at the end with zero  
 values if necessary.\n \* \n \* - If [newSize] is less than the size of the original array, the copy array is truncated to the  
 [newSize].\n \* - If [newSize] is greater than the size of the original array, the extra elements in the copy array are  
 filled with zero values.\n \* \n \* @sample samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n  
 \*^/npublic expect fun FloatArray.copyOfOf(newSize: Int): FloatArray\n/n/\*\*\n \* Returns new array which is a copy of  
 the original array, resized to the given [newSize].\n \* The copy is either truncated or padded at the end with zero  
 values if necessary.\n \* \n \* - If [newSize] is less than the size of the original array, the copy array is truncated to the  
 [newSize].\n \* - If [newSize] is greater than the size of the original array, the extra elements in the copy array are  
 filled with zero values.\n \* \n \* @sample samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n  
 \*^/npublic expect fun DoubleArray.copyOfOf(newSize: Int): DoubleArray\n/n/\*\*\n \* Returns new array which is a  
 copy of the original array, resized to the given [newSize].\n \* The copy is either truncated or padded at the end with  
 `false` values if necessary.\n \* \n \* - If [newSize] is less than the size of the original array, the copy array is  
 truncated to the [newSize].\n \* - If [newSize] is greater than the size of the original array, the extra elements in the  
 copy array are filled with `false` values.\n \* \n \* @sample  
 samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n \*^/npublic expect fun  
 BooleanArray.copyOfOf(newSize: Int): BooleanArray\n/n/\*\*\n \* Returns new array which is a copy of the original  
 array, resized to the given [newSize].\n \* The copy is either truncated or padded at the end with null char (`\u0000`)  
 values if necessary.\n \* \n \* - If [newSize] is less than the size of the original array, the copy array is truncated to the  
 [newSize].\n \* - If [newSize] is greater than the size of the original array, the extra elements in the copy array are  
 filled with null char (`\u0000`) values.\n \* \n \* @sample  
 samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n \*^/npublic expect fun  
 CharArray.copyOfOf(newSize: Int): CharArray\n/n/\*\*\n \* Returns new array which is a copy of the original array,  
 resized to the given [newSize].\n \* The copy is either truncated or padded at the end with `null` values if  
 necessary.\n \* \n \* - If [newSize] is less than the size of the original array, the copy array is truncated to the



specified [element] value.\n \* \n \* @param fromIndex the start of the range (inclusive) to fill, 0 by default.\n \*  
 @param toIndex the end of the range (exclusive) to fill, size of this array by default.\n \* \n \* @throws  
 IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n \*  
 @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n \*  
 @SinceKotlin("1.3")\npublic  
 expect fun <T> Array<T>.fill(element: T, fromIndex: Int = 0, toIndex: Int = size): Unit\n\n/\*\*\n \* Fills this array or  
 its subrange with the specified [element] value.\n \* \n \* @param fromIndex the start of the range (inclusive) to fill,  
 0 by default.\n \* @param toIndex the end of the range (exclusive) to fill, size of this array by default.\n \* \n \*  
 @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this  
 array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n  
 \*/\n@SinceKotlin("1.3")\npublic expect fun ByteArray.fill(element: Byte, fromIndex: Int = 0, toIndex: Int = size):  
 Unit\n\n/\*\*\n \* Fills this array or its subrange with the specified [element] value.\n \* \n \* @param fromIndex the  
 start of the range (inclusive) to fill, 0 by default.\n \* @param toIndex the end of the range (exclusive) to fill, size of  
 this array by default.\n \* \n \* @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is  
 greater than the size of this array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n  
 \*/\n@SinceKotlin("1.3")\npublic expect fun ShortArray.fill(element: Short, fromIndex: Int = 0, toIndex: Int =  
 size): Unit\n\n/\*\*\n \* Fills this array or its subrange with the specified [element] value.\n \* \n \* @param fromIndex  
 the start of the range (inclusive) to fill, 0 by default.\n \* @param toIndex the end of the range (exclusive) to fill, size  
 of this array by default.\n \* \n \* @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex]  
 is greater than the size of this array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than  
 [toIndex].\n \*  
 \*/\n@SinceKotlin("1.3")\npublic expect fun IntArray.fill(element: Int, fromIndex: Int = 0, toIndex: Int =  
 size): Unit\n\n/\*\*\n \* Fills this array or its subrange with the specified [element] value.\n \* \n \* @param  
 fromIndex the start of the range (inclusive) to fill, 0 by default.\n \* @param toIndex the end of the range (exclusive)  
 to fill, size of this array by default.\n \* \n \* @throws IndexOutOfBoundsException if [fromIndex] is less than zero  
 or [toIndex] is greater than the size of this array.\n \* @throws IllegalArgumentException if [fromIndex] is greater  
 than [toIndex].\n \*  
 \*/\n@SinceKotlin("1.3")\npublic expect fun LongArray.fill(element: Long, fromIndex: Int = 0,  
 toIndex: Int = size): Unit\n\n/\*\*\n \* Fills this array or its subrange with the specified [element] value.\n \* \n \*  
 @param fromIndex the start of the range (inclusive) to fill, 0 by default.\n \* @param toIndex the end of the range  
 (exclusive) to fill, size of this array by default.\n \* \n \* @throws IndexOutOfBoundsException if [fromIndex] is less  
 than zero or [toIndex] is greater than the size of this array.\n \* @throws IllegalArgumentException if [fromIndex] is  
 greater than [toIndex].\n \*  
 \*/\n@SinceKotlin("1.3")\npublic expect fun FloatArray.fill(element: Float, fromIndex: Int = 0, toIndex: Int = size): Unit\n\n/\*\*\n \* Fills this array or its subrange with the specified [element] value.\n \* \n \*  
 @param fromIndex the start of the range (inclusive) to fill, 0 by default.\n \* @param toIndex the end of the range  
 (exclusive) to fill, size of this array by default.\n \* \n \* @throws IndexOutOfBoundsException if [fromIndex] is less  
 than zero or [toIndex] is greater than the size of this array.\n \* @throws IllegalArgumentException if [fromIndex] is  
 greater than [toIndex].\n \*  
 \*/\n@SinceKotlin("1.3")\npublic expect fun DoubleArray.fill(element: Double,  
 fromIndex: Int = 0, toIndex: Int = size): Unit\n\n/\*\*\n \* Fills this array or its subrange with the specified [element]  
 value.\n \* \n \* @param fromIndex the start of the range (inclusive) to fill, 0 by default.\n \* @param toIndex the end  
 of the range (exclusive) to fill, size of this array by default.\n \* \n \* @throws IndexOutOfBoundsException if  
 [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n \* @throws  
 IllegalArgumentException if [fromIndex] is greater than [toIndex].\n \*  
 \*/\n@SinceKotlin("1.3")\npublic expect fun BooleanArray.fill(element: Boolean, fromIndex: Int = 0, toIndex: Int = size): Unit\n\n/\*\*\n \* Fills this array or its  
 subrange with the specified [element] value.\n \* \n \* @param fromIndex the start of the range (inclusive) to fill, 0  
 by default.\n \* @param toIndex the end of the range (exclusive) to fill, size of this array by default.\n \* \n \*  
 @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this  
 array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n  
 \*/\n@SinceKotlin("1.3")\npublic expect fun CharArray.fill(element: Char, fromIndex: Int = 0, toIndex: Int = size):  
 Unit\n\n/\*\*\n \* Returns the range of valid indices for the array.\n \*  
 \*/\n@public val <T> Array<out T>.indices:  
 IntRange\n  
 get() = IntRange(0, lastIndex)\n\n/\*\*\n \* Returns the range of valid indices for the array.\n \*  
 \*/\n@public

```

val ByteArray.indices: IntRange\n  get() = IntRange(0, lastIndex)\n\n/**\n * Returns the range of valid indices for
the array.\n */\npublic val ShortArray.indices: IntRange\n  get() = IntRange(0, lastIndex)\n\n/**\n * Returns the
range of valid indices for the array.\n */\npublic val IntArray.indices: IntRange\n  get() = IntRange(0,
lastIndex)\n\n/**\n * Returns the range of valid indices for the array.\n */\npublic val LongArray.indices:
IntRange\n  get() = IntRange(0, lastIndex)\n\n/**\n * Returns the range of valid indices for the array.\n */\npublic
val FloatArray.indices: IntRange\n  get() = IntRange(0, lastIndex)\n\n/**\n * Returns the range of valid indices for
the array.\n */\npublic val DoubleArray.indices: IntRange\n  get() = IntRange(0, lastIndex)\n\n/**\n * Returns the
range of valid indices for the array.\n */\npublic val BooleanArray.indices: IntRange\n  get() = IntRange(0,
lastIndex)\n\n/**\n * Returns the range of valid indices for the array.\n */\npublic val CharArray.indices: IntRange\n
get() = IntRange(0, lastIndex)\n\n/**\n * Returns `true` if the array is empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out T>.isEmpty(): Boolean {\n  return size ==
0\n}\n\n/**\n * Returns `true` if the array is empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun
ByteArray.isEmpty(): Boolean {\n  return size == 0\n}\n\n/**\n * Returns `true` if the array is empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.isEmpty(): Boolean {\n  return size == 0\n}\n\n/**\n * Returns `true` if the array is empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.isEmpty():
Boolean {\n  return size == 0\n}\n\n/**\n * Returns `true` if the array is empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.isEmpty(): Boolean {\n  return size == 0\n}\n\n/**\n * Returns `true` if the array is empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.isEmpty():
Boolean {\n  return size == 0\n}\n\n/**\n * Returns `true` if the array is empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.isEmpty(): Boolean {\n  return size ==
0\n}\n\n/**\n * Returns `true` if the array is empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun
BooleanArray.isEmpty(): Boolean {\n  return size == 0\n}\n\n/**\n * Returns `true` if the array is empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.isEmpty(): Boolean {\n  return size == 0\n}\n\n/**\n * Returns `true` if the array is not empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out
T>.isNotEmpty(): Boolean {\n  return !isEmpty()\n}\n\n/**\n * Returns `true` if the array is not empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.isNotEmpty(): Boolean {\n  return
!isEmpty()\n}\n\n/**\n * Returns `true` if the array is not empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline
fun ShortArray.isNotEmpty(): Boolean {\n  return !isEmpty()\n}\n\n/**\n * Returns `true` if the array is not
empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.isNotEmpty(): Boolean {\n  return
!isEmpty()\n}\n\n/**\n * Returns `true` if the array is not empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline
fun LongArray.isNotEmpty(): Boolean {\n  return !isEmpty()\n}\n\n/**\n * Returns `true` if the array is not
empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.isNotEmpty(): Boolean {\n  return
!isEmpty()\n}\n\n/**\n * Returns `true` if the array is not empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline
fun DoubleArray.isNotEmpty(): Boolean {\n  return !isEmpty()\n}\n\n/**\n * Returns `true` if the array is not
empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.isNotEmpty(): Boolean {\n  return
!isEmpty()\n}\n\n/**\n * Returns `true` if the array is not empty.\n */\n\n@kotlin.internal.InlineOnly\npublic inline
fun CharArray.isNotEmpty(): Boolean {\n  return !isEmpty()\n}\n\n/**\n * Returns the last valid index for the
array.\n */\n\npublic val <T> Array<out T>.lastIndex: Int\n  get() = size - 1\n\n/**\n * Returns the last valid index
for the array.\n */\n\npublic val ByteArray.lastIndex: Int\n  get() = size - 1\n\n/**\n * Returns the last valid index
for the array.\n */\n\npublic val ShortArray.lastIndex: Int\n  get() = size - 1\n\n/**\n * Returns the last valid index
for the array.\n */\n\npublic val IntArray.lastIndex: Int\n  get() = size - 1\n\n/**\n * Returns the last valid index
for the array.\n */\n\npublic val LongArray.lastIndex: Int\n  get() = size - 1\n\n/**\n * Returns the last valid index
for the array.\n */\n\npublic val FloatArray.lastIndex: Int\n  get() = size - 1\n\n/**\n * Returns the last valid index
for the array.\n */\n\npublic val DoubleArray.lastIndex: Int\n  get() = size - 1\n\n/**\n * Returns the last valid index
for the array.\n */\n\npublic val BooleanArray.lastIndex: Int\n  get() = size - 1\n\n/**\n * Returns the last valid index
for the array.\n */\n\npublic val CharArray.lastIndex: Int\n  get() = size - 1\n\n/**\n * Returns an array containing all
elements of the original array and then the given [element].\n */\n\n@Suppress("NO_ACTUAL_FOR_EXPECT")\npublic expect operator fun <T> Array<T>.plus(element: T):

```

`Array<T>.plus(element: T): Array<T>` Returns an array containing all elements of the original array and then the given [element].  
`ByteArray.plus(element: Byte): ByteArray` Returns an array containing all elements of the original array and then the given [element].  
`ShortArray.plus(element: Short): ShortArray` Returns an array containing all elements of the original array and then the given [element].  
`IntArray.plus(element: Int): IntArray` Returns an array containing all elements of the original array and then the given [element].  
`LongArray.plus(element: Long): LongArray` Returns an array containing all elements of the original array and then the given [element].  
`FloatArray.plus(element: Float): FloatArray` Returns an array containing all elements of the original array and then the given [element].  
`DoubleArray.plus(element: Double): DoubleArray` Returns an array containing all elements of the original array and then the given [element].  
`BooleanArray.plus(element: Boolean): BooleanArray` Returns an array containing all elements of the original array and then the given [element].  
`CharArray.plus(element: Char): CharArray` Returns an array containing all elements of the original array and then all elements of the given [elements] collection.  
`Array<T>.plus(elements: Collection<T>): Array<T>` Returns an array containing all elements of the original array and then all elements of the given [elements] collection.  
`ByteArray.plus(elements: Collection<Byte>): ByteArray` Returns an array containing all elements of the original array and then all elements of the given [elements] collection.  
`ShortArray.plus(elements: Collection<Short>): ShortArray` Returns an array containing all elements of the original array and then all elements of the given [elements] collection.  
`IntArray.plus(elements: Collection<Int>): IntArray` Returns an array containing all elements of the original array and then all elements of the given [elements] collection.  
`LongArray.plus(elements: Collection<Long>): LongArray` Returns an array containing all elements of the original array and then all elements of the given [elements] collection.  
`FloatArray.plus(elements: Collection<Float>): FloatArray` Returns an array containing all elements of the original array and then all elements of the given [elements] collection.  
`DoubleArray.plus(elements: Collection<Double>): DoubleArray` Returns an array containing all elements of the original array and then all elements of the given [elements] collection.  
`BooleanArray.plus(elements: Collection<Boolean>): BooleanArray` Returns an array containing all elements of the original array and then all elements of the given [elements] collection.  
`CharArray.plus(elements: Collection<Char>): CharArray` Returns an array containing all elements of the original array and then all elements of the given [elements] array.  
`Array<T>.plus(elements: Array<out T>): Array<T>` Returns an array containing all elements of the original array and then all elements of the given [elements] array.  
`ByteArray.plus(elements: ByteArray): ByteArray` Returns an array containing all elements of the original array and then all elements of the given [elements] array.  
`ShortArray.plus(elements: ShortArray): ShortArray` Returns an array containing all elements of the original array and then all elements of the given [elements] array.  
`IntArray.plus(elements: IntArray): IntArray` Returns an array containing all elements of the original array and then all elements of the given [elements] array.  
`LongArray.plus(elements: LongArray): LongArray` Returns an array containing all elements of the original array and then all elements of the given [elements] array.  
`FloatArray.plus(elements: FloatArray): FloatArray` Returns an array containing all elements of the original array and then all elements of the given [elements] array.  
`DoubleArray.plus(elements: DoubleArray): DoubleArray` Returns an array containing all elements of the original array and then all elements of the given [elements] array.  
`BooleanArray.plus(elements: BooleanArray): BooleanArray` Returns an array containing all elements of

the original array and then all elements of the given [elements] array.

`CharArray.plus(element: CharArray): CharArray` Returns an array containing all elements of the original array and then the given [element].

`Array<T>.plusElement(element: T): Array<T>` Sorts the array in-place.

`samples.collections.Arrays.Sorting.sortArray` Sorts the array in-place.

`IntArray.sort(): Unit` Sorts the array in-place.

`samples.collections.Arrays.Sorting.sortArray` Sorts the array in-place.

`ByteArray.sort(): Unit` Sorts the array in-place.

`samples.collections.Arrays.Sorting.sortArray` Sorts the array in-place.

`ShortArray.sort(): Unit` Sorts the array in-place.

`samples.collections.Arrays.Sorting.sortArray` Sorts the array in-place.

`DoubleArray.sort(): Unit` Sorts the array in-place.

`samples.collections.Arrays.Sorting.sortArray` Sorts the array in-place.

`FloatArray.sort(): Unit` Sorts the array in-place.

`samples.collections.Arrays.Sorting.sortArray` Sorts the array in-place according to the natural order of its elements. The sort is `_stable_`. It means that equal elements preserve their order relative to each other after sorting.

`samples.collections.Arrays.Sorting.sortArrayOfComparable` Sorts a range in the array in-place. The sort is `_stable_`. It means that equal elements preserve their order relative to each other after sorting.

`Comparable<T>> Array<out T>.sort(): Unit` Sorts a range in the array in-place. The sort is `_stable_`. It means that equal elements preserve their order relative to each other after sorting.

`fromIndex` the start of the range (inclusive) to sort, 0 by default.

`toIndex` the end of the range (exclusive) to sort, size of this array by default.

throws `IndexOutOfBoundsException` if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.

throws `IllegalArgumentException` if [fromIndex] is greater than [toIndex].

`samples.collections.Arrays.Sorting.sortRangeOfArrayOfComparable` Sorts a range in the array in-place.

`<T : Comparable<T>> Array<out T>.sort(fromIndex: Int = 0, toIndex: Int = size): Unit` Sorts a range in the array in-place.

`fromIndex` the start of the range (inclusive) to sort, 0 by default.

`toIndex` the end of the range (exclusive) to sort, size of this array by default.

throws `IndexOutOfBoundsException` if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.

throws `IllegalArgumentException` if [fromIndex] is greater than [toIndex].

`samples.collections.Arrays.Sorting.sortRangeOfArray` Sorts a range in the array in-place.

`ByteArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit` Sorts a range in the array in-place.

`fromIndex` the start of the range (inclusive) to sort, 0 by default.

`toIndex` the end of the range (exclusive) to sort, size of this array by default.

throws `IndexOutOfBoundsException` if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.

throws `IllegalArgumentException` if [fromIndex] is greater than [toIndex].

`samples.collections.Arrays.Sorting.sortRangeOfArray` Sorts a range in the array in-place.

`ShortArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit` Sorts a range in the array in-place.

`fromIndex` the start of the range (inclusive) to sort, 0 by default.

`toIndex` the end of the range (exclusive) to sort, size of this array by default.

throws `IndexOutOfBoundsException` if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.

throws `IllegalArgumentException` if [fromIndex] is greater than [toIndex].

`samples.collections.Arrays.Sorting.sortRangeOfArray` Sorts a range in the array in-place.

`IntArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit` Sorts a range in the array in-place.

`fromIndex` the start of the range (inclusive) to sort, 0 by default.

`toIndex` the end of the range (exclusive) to sort, size of this array by default.

throws `IndexOutOfBoundsException` if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.

throws `IllegalArgumentException` if [fromIndex] is greater than [toIndex].

`samples.collections.Arrays.Sorting.sortRangeOfArray` Sorts a range in the array in-place.

`LongArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit` Sorts a range in the array in-place.

`fromIndex` the start of the range (inclusive) to sort, 0 by default.

`toIndex` the end of the range (exclusive) to sort, size of this array by default.

throws

IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n \*  
 @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n \* \n \* @sample  
 samples.collections.Arrays.Sorting.sortRangeOfArray\n \* \n \*@SinceKotlin(\\"1.4\\")\npublic expect fun  
 FloatArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit\n\n/\*\*\n \* Sorts a range in the array in-place.\n \* \n \*\n \* @param fromIndex the start of the range (inclusive) to sort, 0 by default.\n \* @param toIndex the end of the range  
 (exclusive) to sort, size of this array by default.\n \* \n \* @throws IndexOutOfBoundsException if [fromIndex] is  
 less than zero or [toIndex] is greater than the size of this array.\n \* @throws IllegalArgumentException if  
 [fromIndex] is greater than [toIndex].\n \* \n \* @sample samples.collections.Arrays.Sorting.sortRangeOfArray\n\n  
 \* \n \*@SinceKotlin(\\"1.4\\")\npublic expect fun DoubleArray.sort(fromIndex: Int = 0, toIndex: Int = size):  
 Unit\n\n/\*\*\n \* Sorts a range in the array in-place.\n \* \n \* @param fromIndex the start of the range (inclusive) to  
 sort, 0 by default.\n \* @param toIndex the end of the range (exclusive) to sort, size of this array by default.\n \* \n \*\n \* @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this  
 array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n \* \n \* @sample  
 samples.collections.Arrays.Sorting.sortRangeOfArray\n \* \n \*@SinceKotlin(\\"1.4\\")\npublic expect fun  
 CharArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit\n\n/\*\*\n \* Sorts elements of the array in the specified  
 range in-place.\n \* The elements are sorted descending according to their natural sort order.\n \* \n \* The sort is  
 \_stable\_. It means that equal elements preserve their order relative to each other after sorting.\n \* \n \* @param  
 fromIndex the start of the range (inclusive) to sort.\n \* @param toIndex the end of the range (exclusive) to sort.\n \*\n \* \n \* @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of  
 this array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n\n  
 \* \n \*@SinceKotlin(\\"1.4\\")\npublic fun <T : Comparable<T>> Array<out T>.sortDescending(fromIndex: Int,  
 toIndex: Int): Unit {\n \n \n sortWith(reverseOrder(), fromIndex, toIndex)\n }\n\n/\*\*\n \* Sorts elements of the array in  
 the specified range in-place.\n \* The elements are sorted descending according to their natural sort order.\n \* \n \* \n \*  
 @param fromIndex the start of the range (inclusive) to sort.\n \* @param toIndex the end of the range (exclusive) to  
 sort.\n \* \n \* \n \* @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the  
 size of this array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n\n  
 \* \n \*@SinceKotlin(\\"1.4\\")\npublic fun ByteArray.sortDescending(fromIndex: Int, toIndex: Int): Unit {\n\n  
 sort(fromIndex, toIndex)\n \n reverse(fromIndex, toIndex)\n }\n\n/\*\*\n \* Sorts elements of the array in the specified  
 range in-place.\n \* The elements are sorted descending according to their natural sort order.\n \* \n \* @param  
 fromIndex the start of the range (inclusive) to sort.\n \* @param toIndex the end of the range (exclusive) to sort.\n \*\n \* \n \* @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of  
 this array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n\n  
 \* \n \*@SinceKotlin(\\"1.4\\")\npublic fun ShortArray.sortDescending(fromIndex: Int, toIndex: Int): Unit {\n\n  
 sort(fromIndex, toIndex)\n \n reverse(fromIndex, toIndex)\n }\n\n/\*\*\n \* Sorts elements of the array in the specified  
 range in-place.\n \* The elements are sorted descending according to their natural sort order.\n \* \n \* @param  
 fromIndex the start of the range (inclusive) to sort.\n \* @param toIndex the end of the range (exclusive) to sort.\n \*\n \* \n \* @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of  
 this array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n\n  
 \* \n \*@SinceKotlin(\\"1.4\\")\npublic fun IntArray.sortDescending(fromIndex: Int, toIndex: Int): Unit {\n\n  
 sort(fromIndex, toIndex)\n \n reverse(fromIndex, toIndex)\n }\n\n/\*\*\n \* Sorts elements of the array in the specified  
 range in-place.\n \* The elements are sorted descending according to their natural sort order.\n \* \n \* @param  
 fromIndex the start of the range (inclusive) to sort.\n \* @param toIndex the end of the range (exclusive) to sort.\n \*\n \* \n \* @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of  
 this array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n\n  
 \* \n \*@SinceKotlin(\\"1.4\\")\npublic fun LongArray.sortDescending(fromIndex: Int, toIndex: Int): Unit {\n\n  
 sort(fromIndex, toIndex)\n \n reverse(fromIndex, toIndex)\n }\n\n/\*\*\n \* Sorts elements of the array in the specified  
 range in-place.\n \* The elements are sorted descending according to their natural sort order.\n \* \n \* @param  
 fromIndex the start of the range (inclusive) to sort.\n \* @param toIndex the end of the range (exclusive) to sort.\n \*

```

\n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of
this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n
*\n@SinceKotlin("1.4")\npublic fun FloatArray.sortDescending(fromIndex: Int, toIndex: Int): Unit {\n
sort(fromIndex, toIndex)\n reverse(fromIndex, toIndex)\n}\n\n/**\n * Sorts elements of the array in the specified
range in-place.\n * The elements are sorted descending according to their natural sort order.\n * \n * @param
fromIndex the start of the range (inclusive) to sort.\n * @param toIndex the end of the range (exclusive) to sort.\n
*\n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of
this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n
*\n@SinceKotlin("1.4")\npublic fun DoubleArray.sortDescending(fromIndex: Int, toIndex: Int): Unit {\n
sort(fromIndex, toIndex)\n reverse(fromIndex, toIndex)\n}\n\n/**\n * Sorts elements of the array in the specified
range in-place.\n * The elements are sorted descending according to their natural sort order.\n * \n * @param
fromIndex the start of the range (inclusive) to sort.\n * @param toIndex the end of the range (exclusive) to sort.\n
*\n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of
this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n
*\n@SinceKotlin("1.4")\npublic fun CharArray.sortDescending(fromIndex: Int, toIndex: Int): Unit {\n
sort(fromIndex, toIndex)\n reverse(fromIndex, toIndex)\n}\n\n/**\n * Sorts the array in-place according to the
order specified by the given [comparator].\n * \n * The sort is _stable_. It means that equal elements preserve their
order relative to each other after sorting.\n *\npublic expect fun <T> Array<out T>.sortWith(comparator:
Comparator<in T>): Unit\n\n/**\n * Sorts a range in the array in-place with the given [comparator].\n * \n * The
sort is _stable_. It means that equal elements preserve their order relative to each other after sorting.\n
*\n * @param fromIndex the start of the range (inclusive) to sort, 0 by default.\n * @param toIndex the end of the range
(exclusive) to sort, size of this array by default.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is
less than zero or [toIndex] is greater than the size of this array.\n * @throws IllegalArgumentException if
[fromIndex] is greater than [toIndex].\n *\npublic expect fun <T> Array<out T>.sortWith(comparator:
Comparator<in T>, fromIndex: Int = 0, toIndex: Int = size): Unit\n\n/**\n * Returns an array of Boolean containing
all of the elements of this generic array.\n *\npublic fun Array<out Boolean>.toBooleanArray(): BooleanArray {\n
return BooleanArray(size) { index -> this[index] }\n}\n\n/**\n * Returns an array of Byte containing all of the
elements of this generic array.\n *\npublic fun Array<out Byte>.toByteArray(): ByteArray {\n
return ByteArray(size) { index -> this[index] }\n}\n\n/**\n * Returns an array of Char containing all of the elements of this
generic array.\n *\npublic fun Array<out Char>.toCharArray(): CharArray {\n
return CharArray(size) { index -> this[index] }\n}\n\n/**\n * Returns an array of Double containing all of the elements of this generic array.\n
*\npublic fun Array<out Double>.toDoubleArray(): DoubleArray {\n
return DoubleArray(size) { index -> this[index] }\n}\n\n/**\n * Returns an array of Float containing all of the elements of this generic array.\n
*\npublic fun Array<out Float>.toFloatArray(): FloatArray {\n
return FloatArray(size) { index -> this[index] }\n}\n\n/**\n * Returns an array of Int containing all of the elements of this generic array.\n
*\npublic fun Array<out Int>.toIntArray(): IntArray {\n
return IntArray(size) { index -> this[index] }\n}\n\n/**\n * Returns an array of Long containing all of the elements of this generic array.\n
*\npublic fun Array<out Long>.toLongArray(): LongArray {\n
return LongArray(size) { index -> this[index] }\n}\n\n/**\n * Returns an array of Short containing all of the elements of this generic array.\n
*\npublic fun Array<out Short>.toShortArray(): ShortArray {\n
return ShortArray(size) { index -> this[index] }\n}\n\n/**\n * Returns a *typed* object array containing all of the elements
of this primitive array.\n *\npublic expect fun ByteArray.toTypedArray(): Array<Byte>\n\n/**\n * Returns a
*typed* object array containing all of the elements of this primitive array.\n *\npublic expect fun
ShortArray.toTypedArray(): Array<Short>\n\n/**\n * Returns a *typed* object array containing all of the elements
of this primitive array.\n *\npublic expect fun IntArray.toTypedArray(): Array<Int>\n\n/**\n * Returns a *typed*
object array containing all of the elements of this primitive array.\n *\npublic expect fun
LongArray.toTypedArray(): Array<Long>\n\n/**\n * Returns a *typed* object array containing all of the elements
of this primitive array.\n *\npublic expect fun FloatArray.toTypedArray(): Array<Float>\n\n/**\n * Returns a
*typed* object array containing all of the elements of this primitive array.\n *\npublic expect fun

```

DoubleArray.toArray(): Array<Double>\n\n\*\*\n \* Returns a \*typed\* object array containing all of the elements of this primitive array.\n \*^\npublic expect fun BooleanArray.toArray(): Array<Boolean>\n\n\*\*\n \* Returns a \*typed\* object array containing all of the elements of this primitive array.\n \*^\npublic expect fun CharArray.toArray(): Array<Char>\n\n\*\*\n \* Returns a [Map] containing key-value pairs provided by [transform] function\n \* applied to elements of the given array.\n \* \n \* If any of two pairs would have the same key the last one gets added to the map.\n \* \n \* The returned map preserves the entry iteration order of the original array.\n \* \n \* @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitives\n \*^\npublic inline fun <T, K, V> Array<out T>.associate(transform: (T) -> Pair<K, V>): Map<K, V> {\n val capacity = mapCapacity(size).coerceAtLeast(16)\n return associateTo(LinkedHashMap<K, V>(capacity), transform)\n}\n\n\*\*\n \* Returns a [Map] containing key-value pairs provided by [transform] function\n \* applied to elements of the given array.\n \* \n \* If any of two pairs would have the same key the last one gets added to the map.\n \* \n \* The returned map preserves the entry iteration order of the original array.\n \* \n \* @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitives\n \*^\npublic inline fun <K, V> ByteArray.associate(transform: (Byte) -> Pair<K, V>): Map<K, V> {\n val capacity = mapCapacity(size).coerceAtLeast(16)\n return associateTo(LinkedHashMap<K, V>(capacity), transform)\n}\n\n\*\*\n \* Returns a [Map] containing key-value pairs provided by [transform] function\n \* applied to elements of the given array.\n \* \n \* If any of two pairs would have the same key the last one gets added to the map.\n \* \n \* The returned map preserves the entry iteration order of the original array.\n \* \n \* @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitives\n \*^\npublic inline fun <K, V> ShortArray.associate(transform: (Short) -> Pair<K, V>): Map<K, V> {\n val capacity = mapCapacity(size).coerceAtLeast(16)\n return associateTo(LinkedHashMap<K, V>(capacity), transform)\n}\n\n\*\*\n \* Returns a [Map] containing key-value pairs provided by [transform] function\n \* applied to elements of the given array.\n \* \n \* If any of two pairs would have the same key the last one gets added to the map.\n \* \n \* The returned map preserves the entry iteration order of the original array.\n \* \n \* @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitives\n \*^\npublic inline fun <K, V> IntArray.associate(transform: (Int) -> Pair<K, V>): Map<K, V> {\n val capacity = mapCapacity(size).coerceAtLeast(16)\n return associateTo(LinkedHashMap<K, V>(capacity), transform)\n}\n\n\*\*\n \* Returns a [Map] containing key-value pairs provided by [transform] function\n \* applied to elements of the given array.\n \* \n \* If any of two pairs would have the same key the last one gets added to the map.\n \* \n \* The returned map preserves the entry iteration order of the original array.\n \* \n \* @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitives\n \*^\npublic inline fun <K, V> LongArray.associate(transform: (Long) -> Pair<K, V>): Map<K, V> {\n val capacity = mapCapacity(size).coerceAtLeast(16)\n return associateTo(LinkedHashMap<K, V>(capacity), transform)\n}\n\n\*\*\n \* Returns a [Map] containing key-value pairs provided by [transform] function\n \* applied to elements of the given array.\n \* \n \* If any of two pairs would have the same key the last one gets added to the map.\n \* \n \* The returned map preserves the entry iteration order of the original array.\n \* \n \* @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitives\n \*^\npublic inline fun <K, V> FloatArray.associate(transform: (Float) -> Pair<K, V>): Map<K, V> {\n val capacity = mapCapacity(size).coerceAtLeast(16)\n return associateTo(LinkedHashMap<K, V>(capacity), transform)\n}\n\n\*\*\n \* Returns a [Map] containing key-value pairs provided by [transform] function\n \* applied to elements of the given array.\n \* \n \* If any of two pairs would have the same key the last one gets added to the map.\n \* \n \* The returned map preserves the entry iteration order of the original array.\n \* \n \* @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitives\n \*^\npublic inline fun <K, V> DoubleArray.associate(transform: (Double) -> Pair<K, V>): Map<K, V> {\n val capacity = mapCapacity(size).coerceAtLeast(16)\n return associateTo(LinkedHashMap<K, V>(capacity), transform)\n}\n\n\*\*\n \* Returns a [Map] containing key-value pairs provided by [transform] function\n \* applied to elements of the given array.\n \* \n \* If any of two pairs would have the same key the last one gets added to the map.\n \* \n \* The returned map preserves the entry iteration order of the original array.\n \* \n \* @sample

```

samples.collections.Arrays.Transformations.associateArrayOfPrimitives\n *\npublic inline fun <K, V>
BooleanArray.associate(transform: (Boolean) -> Pair<K, V>): Map<K, V> {\n    val capacity =
mapCapacity(size).coerceAtLeast(16)\n    return associateTo(LinkedHashMap<K, V>(capacity),
transform)\n}\n\n/**\n * Returns a [Map] containing key-value pairs provided by [transform] function\n * applied to
elements of the given array.\n * \n * If any of two pairs would have the same key the last one gets added to the
map.\n * \n * The returned map preserves the entry iteration order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitives\n *\npublic inline fun <K, V>
CharArray.associate(transform: (Char) -> Pair<K, V>): Map<K, V> {\n    val capacity =
mapCapacity(size).coerceAtLeast(16)\n    return associateTo(LinkedHashMap<K, V>(capacity),
transform)\n}\n\n/**\n * Returns a [Map] containing the elements from the given array indexed by the key\n *
returned from [keySelector] function applied to each element.\n * \n * If any two elements would have the same key
returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves the entry iteration
order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesBy\n *\npublic inline fun <T, K>
Array<out T>.associateBy(keySelector: (T) -> K): Map<K, T> {\n    val capacity =
mapCapacity(size).coerceAtLeast(16)\n    return associateByTo(LinkedHashMap<K, T>(capacity),
keySelector)\n}\n\n/**\n * Returns a [Map] containing the elements from the given array indexed by the key\n *
returned from [keySelector] function applied to each element.\n * \n * If any two elements would have the same key
returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves the entry iteration
order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesBy\n *\npublic inline fun <K>
ByteArray.associateBy(keySelector: (Byte) -> K): Map<K, Byte> {\n    val capacity =
mapCapacity(size).coerceAtLeast(16)\n    return associateByTo(LinkedHashMap<K, Byte>(capacity),
keySelector)\n}\n\n/**\n * Returns a [Map] containing the elements from the given array indexed by the key\n *
returned from [keySelector] function applied to each element.\n * \n * If any two elements would have the same key
returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves the entry iteration
order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesBy\n *\npublic inline fun <K>
ShortArray.associateBy(keySelector: (Short) -> K): Map<K, Short> {\n    val capacity =
mapCapacity(size).coerceAtLeast(16)\n    return associateByTo(LinkedHashMap<K, Short>(capacity),
keySelector)\n}\n\n/**\n * Returns a [Map] containing the elements from the given array indexed by the key\n *
returned from [keySelector] function applied to each element.\n * \n * If any two elements would have the same key
returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves the entry iteration
order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesBy\n *\npublic inline fun <K>
IntArray.associateBy(keySelector: (Int) -> K): Map<K, Int> {\n    val capacity =
mapCapacity(size).coerceAtLeast(16)\n    return associateByTo(LinkedHashMap<K, Int>(capacity),
keySelector)\n}\n\n/**\n * Returns a [Map] containing the elements from the given array indexed by the key\n *
returned from [keySelector] function applied to each element.\n * \n * If any two elements would have the same key
returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves the entry iteration
order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesBy\n *\npublic inline fun <K>
LongArray.associateBy(keySelector: (Long) -> K): Map<K, Long> {\n    val capacity =
mapCapacity(size).coerceAtLeast(16)\n    return associateByTo(LinkedHashMap<K, Long>(capacity),
keySelector)\n}\n\n/**\n * Returns a [Map] containing the elements from the given array indexed by the key\n *
returned from [keySelector] function applied to each element.\n * \n * If any two elements would have the same key
returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves the entry iteration
order of the original array.\n * \n * @sample

```

```

samples.collections.Arrays.Transformations.associateArrayOfPrimitivesBy\n *\npublic inline fun <K>
FloatArray.associateBy(keySelector: (Float) -> K): Map<K, Float> {\n  val capacity =
mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, Float>(capacity),
keySelector)\n}\n\n/**\n * Returns a [Map] containing the elements from the given array indexed by the key\n *
returned from [keySelector] function applied to each element.\n * \n * If any two elements would have the same key
returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves the entry iteration
order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesBy\n *\npublic inline fun <K>
DoubleArray.associateBy(keySelector: (Double) -> K): Map<K, Double> {\n  val capacity =
mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, Double>(capacity),
keySelector)\n}\n\n/**\n * Returns a [Map] containing the elements from the given array indexed by the key\n *
returned from [keySelector] function applied to each element.\n * \n * If any two elements would have the same key
returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves the entry iteration
order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesBy\n *\npublic inline fun <K>
BooleanArray.associateBy(keySelector: (Boolean) -> K): Map<K, Boolean> {\n  val capacity =
mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, Boolean>(capacity),
keySelector)\n}\n\n/**\n * Returns a [Map] containing the elements from the given array indexed by the key\n *
returned from [keySelector] function applied to each element.\n * \n * If any two elements would have the same key
returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves the entry iteration
order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesBy\n *\npublic inline fun <K>
CharArray.associateBy(keySelector: (Char) -> K): Map<K, Char> {\n  val capacity =
mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, Char>(capacity),
keySelector)\n}\n\n/**\n * Returns a [Map] containing the values provided by [valueTransform] and indexed by
[keySelector] functions applied to elements of the given array.\n * \n * If any two elements would have the same key
returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves the entry iteration
order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByWithValueTransform\n *\npublic inline
fun <T, K, V> Array<out T>.associateBy(keySelector: (T) -> K, valueTransform: (T) -> V): Map<K, V> {\n  val
capacity = mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, V>(capacity),
keySelector, valueTransform)\n}\n\n/**\n * Returns a [Map] containing the values provided by [valueTransform]
and indexed by [keySelector] functions applied to elements of the given array.\n * \n * If any two elements would
have the same key returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves
the entry iteration order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByWithValueTransform\n *\npublic inline
fun <K, V> ByteArray.associateBy(keySelector: (Byte) -> K, valueTransform: (Byte) -> V): Map<K, V> {\n  val
capacity = mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, V>(capacity),
keySelector, valueTransform)\n}\n\n/**\n * Returns a [Map] containing the values provided by [valueTransform]
and indexed by [keySelector] functions applied to elements of the given array.\n * \n * If any two elements would
have the same key returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves
the entry iteration order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByWithValueTransform\n *\npublic inline
fun <K, V> ShortArray.associateBy(keySelector: (Short) -> K, valueTransform: (Short) -> V): Map<K, V> {\n  val
capacity = mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, V>(capacity),
keySelector, valueTransform)\n}\n\n/**\n * Returns a [Map] containing the values provided by [valueTransform]
and indexed by [keySelector] functions applied to elements of the given array.\n * \n * If any two elements would
have the same key returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves
the entry iteration order of the original array.\n * \n * @sample

```

the entry iteration order of the original array.\n \* \n \* @sample

```

samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByWithValueTransform\n *\npublic inline
fun <K, V> IntArray.associateBy(keySelector: (Int) -> K, valueTransform: (Int) -> V): Map<K, V> {\n  val
capacity = mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, V>(capacity),
keySelector, valueTransform)\n}\n\n/**\n * Returns a [Map] containing the values provided by [valueTransform]
and indexed by [keySelector] functions applied to elements of the given array.\n * \n * If any two elements would
have the same key returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves
the entry iteration order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByWithValueTransform\n *\npublic inline
fun <K, V> LongArray.associateBy(keySelector: (Long) -> K, valueTransform: (Long) -> V): Map<K, V> {\n  val
capacity = mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, V>(capacity),
keySelector, valueTransform)\n}\n\n/**\n * Returns a [Map] containing the values provided by [valueTransform]
and indexed by [keySelector] functions applied to elements of the given array.\n * \n * If any two elements would
have the same key returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves
the entry iteration order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByWithValueTransform\n *\npublic inline
fun <K, V> FloatArray.associateBy(keySelector: (Float) -> K, valueTransform: (Float) -> V): Map<K, V> {\n  val
capacity = mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, V>(capacity),
keySelector, valueTransform)\n}\n\n/**\n * Returns a [Map] containing the values provided by [valueTransform]
and indexed by [keySelector] functions applied to elements of the given array.\n * \n * If any two elements would
have the same key returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves
the entry iteration order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByWithValueTransform\n *\npublic inline
fun <K, V> DoubleArray.associateBy(keySelector: (Double) -> K, valueTransform: (Double) -> V): Map<K, V>
{\n  val capacity = mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K,
V>(capacity), keySelector, valueTransform)\n}\n\n/**\n * Returns a [Map] containing the values provided by
[valueTransform] and indexed by [keySelector] functions applied to elements of the given array.\n * \n * If any two
elements would have the same key returned by [keySelector] the last one gets added to the map.\n * \n * The
returned map preserves the entry iteration order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByWithValueTransform\n *\npublic inline
fun <K, V> BooleanArray.associateBy(keySelector: (Boolean) -> K, valueTransform: (Boolean) -> V): Map<K, V>
{\n  val capacity = mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K,
V>(capacity), keySelector, valueTransform)\n}\n\n/**\n * Returns a [Map] containing the values provided by
[valueTransform] and indexed by [keySelector] functions applied to elements of the given array.\n * \n * If any two
elements would have the same key returned by [keySelector] the last one gets added to the map.\n * \n * The
returned map preserves the entry iteration order of the original array.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByWithValueTransform\n *\npublic inline
fun <K, V> CharArray.associateBy(keySelector: (Char) -> K, valueTransform: (Char) -> V): Map<K, V> {\n  val
capacity = mapCapacity(size).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, V>(capacity),
keySelector, valueTransform)\n}\n\n/**\n * Populates and returns the [destination] mutable map with key-value
pairs,\n * where key is provided by the [keySelector] function applied to each element of the given array\n * and
value is the element itself.\n * \n * If any two elements would have the same key returned by [keySelector] the last
one gets added to the map.\n * \n * @sample
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByTo\n *\npublic inline fun <T, K, M :
MutableMap<in K, in T>> Array<out T>.associateByTo(destination: M, keySelector: (T) -> K): M {\n  for
(element in this) {\n    destination.put(keySelector(element), element)\n  }\n  return destination\n}\n\n/**\n *
Populates and returns the [destination] mutable map with key-value pairs,\n * where key is provided by the
[keySelector] function applied to each element of the given array\n * and value is the element itself.\n * \n * If any

```

two elements would have the same key returned by [keySelector] the last one gets added to the map.

```

@sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByTo<K,
M : MutableMap<in K, in Byte>> ByteArray.associateByTo(destination: M, keySelector: (Byte) -> K): M {
  for (element in this) {
    destination.put(keySelector(element), element)
  }
  return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs, where key is provided by the [keySelector] function applied to each element of the given array and value is the element itself. If any two elements would have the same key returned by [keySelector] the last one gets added to the map.

```

@sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByTo<K,
M : MutableMap<in K, in Short>> ShortArray.associateByTo(destination: M, keySelector: (Short) -> K): M {
  for (element in this) {
    destination.put(keySelector(element), element)
  }
  return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs, where key is provided by the [keySelector] function applied to each element of the given array and value is the element itself. If any two elements would have the same key returned by [keySelector] the last one gets added to the map.

```

@sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByTo<K,
M : MutableMap<in K, in Int>> IntArray.associateByTo(destination: M, keySelector: (Int) -> K): M {
  for (element in this) {
    destination.put(keySelector(element), element)
  }
  return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs, where key is provided by the [keySelector] function applied to each element of the given array and value is the element itself. If any two elements would have the same key returned by [keySelector] the last one gets added to the map.

```

@sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByTo<K,
M : MutableMap<in K, in Long>> LongArray.associateByTo(destination: M, keySelector: (Long) -> K): M {
  for (element in this) {
    destination.put(keySelector(element), element)
  }
  return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs, where key is provided by the [keySelector] function applied to each element of the given array and value is the element itself. If any two elements would have the same key returned by [keySelector] the last one gets added to the map.

```

@sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByTo<K,
M : MutableMap<in K, in Float>> FloatArray.associateByTo(destination: M, keySelector: (Float) -> K): M {
  for (element in this) {
    destination.put(keySelector(element), element)
  }
  return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs, where key is provided by the [keySelector] function applied to each element of the given array and value is the element itself. If any two elements would have the same key returned by [keySelector] the last one gets added to the map.

```

@sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByTo<K,
M : MutableMap<in K, in Double>> DoubleArray.associateByTo(destination: M, keySelector: (Double) -> K): M {
  for (element in this) {
    destination.put(keySelector(element), element)
  }
  return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs, where key is provided by the [keySelector] function applied to each element of the given array and value is the element itself. If any two elements would have the same key returned by [keySelector] the last one gets added to the map.

```

@sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByTo<K,
M : MutableMap<in K, in Boolean>> BooleanArray.associateByTo(destination: M, keySelector:
(Boolean) -> K): M {
  for (element in this) {
    destination.put(keySelector(element), element)
  }
  return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs, where key is provided by the [keySelector] function applied to each element of the given array and value is the element itself. If any two elements would have the same key returned by [keySelector] the last one gets added to the map.

```

@sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByTo<K,
M : MutableMap<in K, in Char>> CharArray.associateByTo(destination: M, keySelector: (Char) ->
K): M {
  for (element in this) {
    destination.put(keySelector(element), element)
  }
  return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs, where key is provided by the [keySelector] function and value is provided by the [valueTransform] function applied to

elements of the given array.  
\* If any two elements would have the same key returned by [keySelector] the last one gets added to the map.  
\* @sample

```
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByToWithValueTransform  
inline fun <T, K, V, M : MutableMap<in K, in V>> Array<out T>.associateByTo(destination: M, keySelector: (T) -> K, valueTransform: (T) -> V): M {  
    for (element in this) {  
        destination.put(keySelector(element), valueTransform(element))  
    }  
    return destination  
}  
* Populates and returns the [destination] mutable map with key-value pairs,  
* where key is provided by the [keySelector] function and  
* and value is provided by the [valueTransform] function applied to elements of the given array.  
* If any two elements would have the same key returned by [keySelector] the last one gets added to the map.  
* @sample
```

```
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByToWithValueTransform  
inline fun <K, V, M : MutableMap<in K, in V>> ByteArray.associateByTo(destination: M, keySelector: (Byte) -> K, valueTransform: (Byte) -> V): M {  
    for (element in this) {  
        destination.put(keySelector(element), valueTransform(element))  
    }  
    return destination  
}  
* Populates and returns the [destination] mutable map with key-value pairs,  
* where key is provided by the [keySelector] function and  
* and value is provided by the [valueTransform] function applied to elements of the given array.  
* If any two elements would have the same key returned by [keySelector] the last one gets added to the map.  
* @sample
```

```
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByToWithValueTransform  
inline fun <K, V, M : MutableMap<in K, in V>> ShortArray.associateByTo(destination: M, keySelector: (Short) -> K, valueTransform: (Short) -> V): M {  
    for (element in this) {  
        destination.put(keySelector(element), valueTransform(element))  
    }  
    return destination  
}  
* Populates and returns the [destination] mutable map with key-value pairs,  
* where key is provided by the [keySelector] function and  
* and value is provided by the [valueTransform] function applied to elements of the given array.  
* If any two elements would have the same key returned by [keySelector] the last one gets added to the map.  
* @sample
```

```
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByToWithValueTransform  
inline fun <K, V, M : MutableMap<in K, in V>> IntArray.associateByTo(destination: M, keySelector: (Int) -> K, valueTransform: (Int) -> V): M {  
    for (element in this) {  
        destination.put(keySelector(element), valueTransform(element))  
    }  
    return destination  
}  
* Populates and returns the [destination] mutable map with key-value pairs,  
* where key is provided by the [keySelector] function and  
* and value is provided by the [valueTransform] function applied to elements of the given array.  
* If any two elements would have the same key returned by [keySelector] the last one gets added to the map.  
* @sample
```

```
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByToWithValueTransform  
inline fun <K, V, M : MutableMap<in K, in V>> LongArray.associateByTo(destination: M, keySelector: (Long) -> K, valueTransform: (Long) -> V): M {  
    for (element in this) {  
        destination.put(keySelector(element), valueTransform(element))  
    }  
    return destination  
}  
* Populates and returns the [destination] mutable map with key-value pairs,  
* where key is provided by the [keySelector] function and  
* and value is provided by the [valueTransform] function applied to elements of the given array.  
* If any two elements would have the same key returned by [keySelector] the last one gets added to the map.  
* @sample
```

```
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByToWithValueTransform  
inline fun <K, V, M : MutableMap<in K, in V>> FloatArray.associateByTo(destination: M, keySelector: (Float) -> K, valueTransform: (Float) -> V): M {  
    for (element in this) {  
        destination.put(keySelector(element), valueTransform(element))  
    }  
    return destination  
}  
* Populates and returns the [destination] mutable map with key-value pairs,  
* where key is provided by the [keySelector] function and  
* and value is provided by the [valueTransform] function applied to elements of the given array.  
* If any two elements would have the same key returned by [keySelector] the last one gets added to the map.  
* @sample
```

```
samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByToWithValueTransform  
inline fun <K, V, M : MutableMap<in K, in V>> DoubleArray.associateByTo(destination: M, keySelector: (Double) -> K, valueTransform: (Double) -> V): M {  
    for (element in this) {  
        destination.put(keySelector(element), valueTransform(element))  
    }  
    return destination  
}  
* Populates
```

and returns the [destination] mutable map with key-value pairs, where key is provided by the [keySelector] function and value is provided by the [valueTransform] function applied to elements of the given array. If any two elements would have the same key returned by [keySelector] the last one gets added to the map. @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByToWithValueTransform

```

public inline fun <K, V, M : MutableMap<in K, in V>> BooleanArray.associateByTo(destination: M,
keySelector: (Boolean) -> K, valueTransform: (Boolean) -> V): M {
    for (element in this) {
        destination.put(keySelector(element), valueTransform(element))
    }
    return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs, where key is provided by the [keySelector] function and value is provided by the [valueTransform] function applied to elements of the given array. If any two elements would have the same key returned by [keySelector] the last one gets added to the map. @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesByToWithValueTransform

```

public inline fun <K, V, M : MutableMap<in K, in V>> CharArray.associateByTo(destination: M, keySelector:
(Char) -> K, valueTransform: (Char) -> V): M {
    for (element in this) {
        destination.put(keySelector(element), valueTransform(element))
    }
    return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs provided by [transform] function applied to each element of the given array. If any of two pairs would have the same key the last one gets added to the map. @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesTo

```

public inline fun <T, K, V, M : MutableMap<in K, in V>> Array<out T>.associateTo(destination: M, transform: (T) ->
Pair<K, V>): M {
    for (element in this) {
        destination += transform(element)
    }
    return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs provided by [transform] function applied to each element of the given array. If any of two pairs would have the same key the last one gets added to the map. @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesTo

```

public inline fun <K, V, M : MutableMap<in K, in V>> ByteArray.associateTo(destination: M, transform: (Byte) -> Pair<K, V>): M {
    for (element in this) {
        destination += transform(element)
    }
    return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs provided by [transform] function applied to each element of the given array. If any of two pairs would have the same key the last one gets added to the map. @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesTo

```

public inline fun <K, V, M : MutableMap<in K, in V>> ShortArray.associateTo(destination: M, transform: (Short) ->
Pair<K, V>): M {
    for (element in this) {
        destination += transform(element)
    }
    return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs provided by [transform] function applied to each element of the given array. If any of two pairs would have the same key the last one gets added to the map. @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesTo

```

public inline fun <K, V, M : MutableMap<in K, in V>> IntArray.associateTo(destination: M, transform: (Int) -> Pair<K, V>): M {
    for (element in this) {
        destination += transform(element)
    }
    return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs provided by [transform] function applied to each element of the given array. If any of two pairs would have the same key the last one gets added to the map. @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesTo

```

public inline fun <K, V, M : MutableMap<in K, in V>> LongArray.associateTo(destination: M, transform: (Long) ->
Pair<K, V>): M {
    for (element in this) {
        destination += transform(element)
    }
    return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs provided by [transform] function applied to each element of the given array. If any of two pairs would have the same key the last one gets added to the map. @sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesTo

```

public inline fun <K, V, M : MutableMap<in K, in V>> FloatArray.associateTo(destination: M, transform: (Float) -> Pair<K, V>): M {
    for (element in this) {
        destination += transform(element)
    }
    return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs provided by [transform] function applied to each

element of the given array.

If any of two pairs would have the same key the last one gets added to the map.

`@sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesTo`

```

public inline fun <K, V, M : MutableMap<in K, in V>> DoubleArray.associateTo(destination: M, transform: (Double) -> Pair<K, V>): M {
    for (element in this) {
        destination += transform(element)
    }
    return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs provided by [transform] function applied to each element of the given array.

If any of two pairs would have the same key the last one gets added to the map.

`@sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesTo`

```

public inline fun <K, V, M : MutableMap<in K, in V>> BooleanArray.associateTo(destination: M, transform: (Boolean) -> Pair<K, V>): M {
    for (element in this) {
        destination += transform(element)
    }
    return destination
}

```

Populates and returns the [destination] mutable map with key-value pairs provided by [transform] function applied to each element of the given array.

If any of two pairs would have the same key the last one gets added to the map.

`@sample samples.collections.Arrays.Transformations.associateArrayOfPrimitivesTo`

```

public inline fun <K, V, M : MutableMap<in K, in V>> CharArray.associateTo(destination: M, transform: (Char) -> Pair<K, V>): M {
    for (element in this) {
        destination += transform(element)
    }
    return destination
}

```

Returns a [Map] where keys are elements from the given array and values are produced by the [valueSelector] function applied to each element.

If any two elements are equal, the last one gets added to the map.

The returned map preserves the entry iteration order of the original array.

`@sample samples.collections.Collections.Transformations.associateWith`

```

@SinceKotlin("1.4")
public inline fun <K, V> Array<out K>.associateWith(valueSelector: (K) -> V): Map<K, V> {
    val result =
        LinkedHashMap<K, V>(mapCapacity(size).coerceAtLeast(16))
    return associateWithTo(result, valueSelector)
}

```

Returns a [Map] where keys are elements from the given array and values are produced by the [valueSelector] function applied to each element.

If any two elements are equal, the last one gets added to the map.

The returned map preserves the entry iteration order of the original array.

`@sample samples.collections.Collections.Transformations.associateWith`

```

@SinceKotlin("1.4")
@kotlin.internal.InlineOnly
public inline fun <V>
ByteArray.associateWith(valueSelector: (Byte) -> V): Map<Byte, V> {
    val result = LinkedHashMap<Byte, V>(mapCapacity(size).coerceAtLeast(16))
    return associateWithTo(result, valueSelector)
}

```

Returns a [Map] where keys are elements from the given array and values are produced by the [valueSelector] function applied to each element.

If any two elements are equal, the last one gets added to the map.

The returned map preserves the entry iteration order of the original array.

`@sample samples.collections.Collections.Transformations.associateWith`

```

@SinceKotlin("1.4")
@kotlin.internal.InlineOnly
public inline fun <V>
ShortArray.associateWith(valueSelector: (Short) -> V): Map<Short, V> {
    val result = LinkedHashMap<Short, V>(mapCapacity(size).coerceAtLeast(16))
    return associateWithTo(result, valueSelector)
}

```

Returns a [Map] where keys are elements from the given array and values are produced by the [valueSelector] function applied to each element.

If any two elements are equal, the last one gets added to the map.

The returned map preserves the entry iteration order of the original array.

`@sample samples.collections.Collections.Transformations.associateWith`

```

@SinceKotlin("1.4")
@kotlin.internal.InlineOnly
public inline fun <V>
IntArray.associateWith(valueSelector: (Int) -> V): Map<Int, V> {
    val result = LinkedHashMap<Int, V>(mapCapacity(size).coerceAtLeast(16))
    return associateWithTo(result, valueSelector)
}

```

Returns a [Map] where keys are elements from the given array and values are produced by the [valueSelector] function applied to each element.

If any two elements are equal, the last one gets added to the map.

The returned map preserves the entry iteration order of the original array.

`@sample samples.collections.Collections.Transformations.associateWith`

```

@SinceKotlin("1.4")
@kotlin.internal.InlineOnly
public inline fun <V>
LongArray.associateWith(valueSelector: (Long) -> V): Map<Long, V> {
    val result = LinkedHashMap<Long, V>
}

```

```

V>(mapCapacity(size).coerceAtLeast(16))\n  return associateWithTo(result, valueSelector)\n}\n\n/**\n * Returns a
[Map] where keys are elements from the given array and values are\n * produced by the [valueSelector] function
applied to each element.\n * \n * If any two elements are equal, the last one gets added to the map.\n * \n * The
returned map preserves the entry iteration order of the original array.\n * \n * @sample
samples.collections.Collections.Transformations.associateWith\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <V>
FloatArray.associateWith(valueSelector: (Float) -> V): Map<Float, V> {\n  val result = LinkedHashMap<Float,
V>(mapCapacity(size).coerceAtLeast(16))\n  return associateWithTo(result, valueSelector)\n}\n\n/**\n * Returns a
[Map] where keys are elements from the given array and values are\n * produced by the [valueSelector] function
applied to each element.\n * \n * If any two elements are equal, the last one gets added to the map.\n * \n * The
returned map preserves the entry iteration order of the original array.\n * \n * @sample
samples.collections.Collections.Transformations.associateWith\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <V>
DoubleArray.associateWith(valueSelector: (Double) -> V): Map<Double, V> {\n  val result =
LinkedHashMap<Double, V>(mapCapacity(size).coerceAtLeast(16))\n  return associateWithTo(result,
valueSelector)\n}\n\n/**\n * Returns a [Map] where keys are elements from the given array and values are\n *
produced by the [valueSelector] function applied to each element.\n * \n * If any two elements are equal, the last one
gets added to the map.\n * \n * The returned map preserves the entry iteration order of the original array.\n * \n *
@sample samples.collections.Collections.Transformations.associateWith\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <V>
BooleanArray.associateWith(valueSelector: (Boolean) -> V): Map<Boolean, V> {\n  val result =
LinkedHashMap<Boolean, V>(mapCapacity(size).coerceAtLeast(16))\n  return associateWithTo(result,
valueSelector)\n}\n\n/**\n * Returns a [Map] where keys are elements from the given array and values are\n *
produced by the [valueSelector] function applied to each element.\n * \n * If any two elements are equal, the last one
gets added to the map.\n * \n * The returned map preserves the entry iteration order of the original array.\n * \n *
@sample samples.collections.Collections.Transformations.associateWith\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <V>
CharArray.associateWith(valueSelector: (Char) -> V): Map<Char, V> {\n  val result = LinkedHashMap<Char,
V>(mapCapacity(size.coerceAtMost(128)).coerceAtLeast(16))\n  return associateWithTo(result,
valueSelector)\n}\n\n/**\n * Populates and returns the [destination] mutable map with key-value pairs for each
element of the given array,\n * where key is the element itself and value is provided by the [valueSelector] function
applied to that key.\n * \n * If any two elements are equal, the last one overwrites the former value in the map.\n * \n
*\n * @sample samples.collections.Collections.Transformations.associateWithTo\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <K, V, M : MutableMap<in K, in V>> Array<out K>.associateWithTo(destination: M, valueSelector: (K)
-> V): M {\n  for (element in this) {\n    destination.put(element, valueSelector(element))\n  }\n  return
destination\n}\n\n/**\n * Populates and returns the [destination] mutable map with key-value pairs for each element
of the given array,\n * where key is the element itself and value is provided by the [valueSelector] function applied
to that key.\n * \n * If any two elements are equal, the last one overwrites the former value in the map.\n * \n *
@sample samples.collections.Collections.Transformations.associateWithTo\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <V, M : MutableMap<in Byte, in V>>
ByteArray.associateWithTo(destination: M, valueSelector: (Byte) -> V): M {\n  for (element in this) {\n
  destination.put(element, valueSelector(element))\n  }\n  return destination\n}\n\n/**\n * Populates and returns the
[destination] mutable map with key-value pairs for each element of the given array,\n * where key is the element
itself and value is provided by the [valueSelector] function applied to that key.\n * \n * If any two elements are
equal, the last one overwrites the former value in the map.\n * \n * @sample
samples.collections.Collections.Transformations.associateWithTo\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <V, M : MutableMap<in Short, in V>>
ShortArray.associateWithTo(destination: M, valueSelector: (Short) -> V): M {\n  for (element in this) {\n

```

```

destination.put(element, valueSelector(element))\n } \n return destination\n}\n\n/**\n * Populates and returns the
[destination] mutable map with key-value pairs for each element of the given array,\n * where key is the element
itself and value is provided by the [valueSelector] function applied to that key.\n * \n * If any two elements are
equal, the last one overwrites the former value in the map.\n * \n * @sample
samples.collections.Collections.Transformations.associateWithTo\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <V, M : MutableMap<in Int, in V>>
IntArray.associateWithTo(destination: M, valueSelector: (Int) -> V): M {\n for (element in this) {\n
destination.put(element, valueSelector(element))\n } \n return destination\n}\n\n/**\n * Populates and returns the
[destination] mutable map with key-value pairs for each element of the given array,\n * where key is the element
itself and value is provided by the [valueSelector] function applied to that key.\n * \n * If any two elements are
equal, the last one overwrites the former value in the map.\n * \n * @sample
samples.collections.Collections.Transformations.associateWithTo\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <V, M : MutableMap<in Long, in V>>
LongArray.associateWithTo(destination: M, valueSelector: (Long) -> V): M {\n for (element in this) {\n
destination.put(element, valueSelector(element))\n } \n return destination\n}\n\n/**\n * Populates and returns the
[destination] mutable map with key-value pairs for each element of the given array,\n * where key is the element
itself and value is provided by the [valueSelector] function applied to that key.\n * \n * If any two elements are
equal, the last one overwrites the former value in the map.\n * \n * @sample
samples.collections.Collections.Transformations.associateWithTo\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <V, M : MutableMap<in Float, in V>>
FloatArray.associateWithTo(destination: M, valueSelector: (Float) -> V): M {\n for (element in this) {\n
destination.put(element, valueSelector(element))\n } \n return destination\n}\n\n/**\n * Populates and returns the
[destination] mutable map with key-value pairs for each element of the given array,\n * where key is the element
itself and value is provided by the [valueSelector] function applied to that key.\n * \n * If any two elements are
equal, the last one overwrites the former value in the map.\n * \n * @sample
samples.collections.Collections.Transformations.associateWithTo\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <V, M : MutableMap<in Double, in V>>
DoubleArray.associateWithTo(destination: M, valueSelector: (Double) -> V): M {\n for (element in this) {\n
destination.put(element, valueSelector(element))\n } \n return destination\n}\n\n/**\n * Populates and returns the
[destination] mutable map with key-value pairs for each element of the given array,\n * where key is the element
itself and value is provided by the [valueSelector] function applied to that key.\n * \n * If any two elements are
equal, the last one overwrites the former value in the map.\n * \n * @sample
samples.collections.Collections.Transformations.associateWithTo\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <V, M : MutableMap<in Boolean, in
V>> BooleanArray.associateWithTo(destination: M, valueSelector: (Boolean) -> V): M {\n for (element in this)
{\n destination.put(element, valueSelector(element))\n } \n return destination\n}\n\n/**\n * Populates and
returns the [destination] mutable map with key-value pairs for each element of the given array,\n * where key is the
element itself and value is provided by the [valueSelector] function applied to that key.\n * \n * If any two elements
are equal, the last one overwrites the former value in the map.\n * \n * @sample
samples.collections.Collections.Transformations.associateWithTo\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <V, M : MutableMap<in Char, in V>>
CharArray.associateWithTo(destination: M, valueSelector: (Char) -> V): M {\n for (element in this) {\n
destination.put(element, valueSelector(element))\n } \n return destination\n}\n\n/**\n * Appends all elements to
the given [destination] collection.\n *\npublic fun <T, C : MutableCollection<in T>> Array<out
T>.toCollection(destination: C): C {\n for (item in this) {\n destination.add(item)\n } \n return
destination\n}\n\n/**\n * Appends all elements to the given [destination] collection.\n *\npublic fun <C :
MutableCollection<in Byte>> ByteArray.toCollection(destination: C): C {\n for (item in this) {\n
destination.add(item)\n } \n return destination\n}\n\n/**\n * Appends all elements to the given [destination]

```

```

collection.\n */\npublic fun <C : MutableCollection<in Short>> ShortArray.toCollection(destination: C): C {\n for
(item in this) {\n destination.add(item)\n }\n return destination\n}\n\n/**\n * Appends all elements to the
given [destination] collection.\n */\npublic fun <C : MutableCollection<in Int>> IntArray.toCollection(destination:
C): C {\n for (item in this) {\n destination.add(item)\n }\n return destination\n}\n\n/**\n * Appends all
elements to the given [destination] collection.\n */\npublic fun <C : MutableCollection<in Long>>
LongArray.toCollection(destination: C): C {\n for (item in this) {\n destination.add(item)\n }\n return
destination\n}\n\n/**\n * Appends all elements to the given [destination] collection.\n */\npublic fun <C :
MutableCollection<in Float>> FloatArray.toCollection(destination: C): C {\n for (item in this) {\n
destination.add(item)\n }\n return destination\n}\n\n/**\n * Appends all elements to the given [destination]
collection.\n */\npublic fun <C : MutableCollection<in Double>> DoubleArray.toCollection(destination: C): C {\n
for (item in this) {\n destination.add(item)\n }\n return destination\n}\n\n/**\n * Appends all elements to the
given [destination] collection.\n */\npublic fun <C : MutableCollection<in Boolean>>
BooleanArray.toCollection(destination: C): C {\n for (item in this) {\n destination.add(item)\n }\n return
destination\n}\n\n/**\n * Appends all elements to the given [destination] collection.\n */\npublic fun <C :
MutableCollection<in Char>> CharArray.toCollection(destination: C): C {\n for (item in this) {\n
destination.add(item)\n }\n return destination\n}\n\n/**\n * Returns a new [HashSet] of all elements.\n
*/\npublic fun <T> Array<out T>.toHashSet(): HashSet<T> {\n return
toCollection(HashSet<T>(mapCapacity(size)))\n}\n\n/**\n * Returns a new [HashSet] of all elements.\n
*/\npublic fun ByteArray.toHashSet(): HashSet<Byte> {\n return
toCollection(HashSet<Byte>(mapCapacity(size)))\n}\n\n/**\n * Returns a new [HashSet] of all elements.\n
*/\npublic fun ShortArray.toHashSet(): HashSet<Short> {\n return
toCollection(HashSet<Short>(mapCapacity(size)))\n}\n\n/**\n * Returns a new [HashSet] of all elements.\n
*/\npublic fun IntArray.toHashSet(): HashSet<Int> {\n return
toCollection(HashSet<Int>(mapCapacity(size)))\n}\n\n/**\n * Returns a new [HashSet] of all elements.\n
*/\npublic fun LongArray.toHashSet(): HashSet<Long> {\n return
toCollection(HashSet<Long>(mapCapacity(size)))\n}\n\n/**\n * Returns a new [HashSet] of all elements.\n
*/\npublic fun FloatArray.toHashSet(): HashSet<Float> {\n return
toCollection(HashSet<Float>(mapCapacity(size)))\n}\n\n/**\n * Returns a new [HashSet] of all elements.\n
*/\npublic fun DoubleArray.toHashSet(): HashSet<Double> {\n return
toCollection(HashSet<Double>(mapCapacity(size)))\n}\n\n/**\n * Returns a new [HashSet] of all elements.\n
*/\npublic fun BooleanArray.toHashSet(): HashSet<Boolean> {\n return
toCollection(HashSet<Boolean>(mapCapacity(size)))\n}\n\n/**\n * Returns a new [HashSet] of all elements.\n
*/\npublic fun CharArray.toHashSet(): HashSet<Char> {\n return
toCollection(HashSet<Char>(mapCapacity(size.coerceAtMost(128))))\n}\n\n/**\n * Returns a [List] containing all
elements.\n */\npublic fun <T> Array<out T>.toList(): List<T> {\n return when (size) {\n 0 -> emptyList()\n
1 -> listOf(this[0])\n else -> this.toMutableList()\n }\n}\n\n/**\n * Returns a [List] containing all
elements.\n */\npublic fun ByteArray.toList(): List<Byte> {\n return when (size) {\n 0 -> emptyList()\n
1 -> listOf(this[0])\n else -> this.toMutableList()\n }\n}\n\n/**\n * Returns a [List] containing all elements.\n
*/\npublic fun ShortArray.toList(): List<Short> {\n return when (size) {\n 0 -> emptyList()\n
1 -> listOf(this[0])\n else -> this.toMutableList()\n }\n}\n\n/**\n * Returns a [List] containing all elements.\n
*/\npublic fun IntArray.toList(): List<Int> {\n return when (size) {\n 0 -> emptyList()\n
1 -> listOf(this[0])\n else -> this.toMutableList()\n }\n}\n\n/**\n * Returns a [List] containing all elements.\n
*/\npublic fun LongArray.toList(): List<Long> {\n return when (size) {\n 0 -> emptyList()\n
1 -> listOf(this[0])\n else -> this.toMutableList()\n }\n}\n\n/**\n * Returns a [List] containing all elements.\n
*/\npublic fun FloatArray.toList(): List<Float> {\n return when (size) {\n 0 -> emptyList()\n
1 -> listOf(this[0])\n else -> this.toMutableList()\n }\n}\n\n/**\n * Returns a [List] containing all elements.\n
*/\npublic fun DoubleArray.toList(): List<Double> {\n return when (size) {\n 0 -> emptyList()\n
1 -> listOf(this[0])\n else -> this.toMutableList()\n }\n}\n\n/**\n * Returns a [List] containing all elements.\n

```

```

*\npublic fun BooleanArray.toList(): List<Boolean> {\n    return when (size) {\n        0 -> emptyList()\n        1 ->
listOf(this[0])\n        else -> this.toMutableList()\n    }\n}\n\n/**\n * Returns a [List] containing all elements.\n */
*\npublic fun CharArray.toList(): List<Char> {\n    return when (size) {\n        0 -> emptyList()\n        1 ->
listOf(this[0])\n        else -> this.toMutableList()\n    }\n}\n\n/**\n * Returns a new [MutableList] filled with all
elements of this array.\n */
*\npublic fun <T> Array<out T>.toMutableList(): MutableList<T> {\n    return
ArrayList(this.asCollection())\n}\n\n/**\n * Returns a new [MutableList] filled with all elements of this array.\n */
*\npublic fun ByteArray.toMutableList(): MutableList<Byte> {\n    val list = ArrayList<Byte>(size)\n    for (item
in this) list.add(item)\n    return list\n}\n\n/**\n * Returns a new [MutableList] filled with all elements of this
array.\n */
*\npublic fun ShortArray.toMutableList(): MutableList<Short> {\n    val list = ArrayList<Short>(size)\n    for (item
in this) list.add(item)\n    return list\n}\n\n/**\n * Returns a new [MutableList] filled with all elements of
this array.\n */
*\npublic fun IntArray.toMutableList(): MutableList<Int> {\n    val list = ArrayList<Int>(size)\n    for (item
in this) list.add(item)\n    return list\n}\n\n/**\n * Returns a new [MutableList] filled with all elements of this
array.\n */
*\npublic fun LongArray.toMutableList(): MutableList<Long> {\n    val list = ArrayList<Long>(size)\n    for (item
in this) list.add(item)\n    return list\n}\n\n/**\n * Returns a new [MutableList] filled with all elements of
this array.\n */
*\npublic fun FloatArray.toMutableList(): MutableList<Float> {\n    val list =
ArrayList<Float>(size)\n    for (item in this) list.add(item)\n    return list\n}\n\n/**\n * Returns a new [MutableList]
filled with all elements of this array.\n */
*\npublic fun DoubleArray.toMutableList(): MutableList<Double> {\n    val
list = ArrayList<Double>(size)\n    for (item in this) list.add(item)\n    return list\n}\n\n/**\n * Returns a new
[MutableList] filled with all elements of this array.\n */
*\npublic fun BooleanArray.toMutableList():
MutableList<Boolean> {\n    val list = ArrayList<Boolean>(size)\n    for (item in this) list.add(item)\n    return
list\n}\n\n/**\n * Returns a new [MutableList] filled with all elements of this array.\n */
*\npublic fun
CharArray.toMutableList(): MutableList<Char> {\n    val list = ArrayList<Char>(size)\n    for (item in this)
list.add(item)\n    return list\n}\n\n/**\n * Returns a [Set] of all elements.\n * \n * The returned set preserves the
element iteration order of the original array.\n */
*\npublic fun <T> Array<out T>.toSet(): Set<T> {\n    return when
(size) {\n        0 -> emptySet()\n        1 -> setOf(this[0])\n        else ->
toCollection(LinkedHashSet<T>(mapCapacity(size)))\n    }\n}\n\n/**\n * Returns a [Set] of all elements.\n * \n *
The returned set preserves the element iteration order of the original array.\n */
*\npublic fun ByteArray.toSet():
Set<Byte> {\n    return when (size) {\n        0 -> emptySet()\n        1 -> setOf(this[0])\n        else ->
toCollection(LinkedHashSet<Byte>(mapCapacity(size)))\n    }\n}\n\n/**\n * Returns a [Set] of all elements.\n * \n *
The returned set preserves the element iteration order of the original array.\n */
*\npublic fun ShortArray.toSet():
Set<Short> {\n    return when (size) {\n        0 -> emptySet()\n        1 -> setOf(this[0])\n        else ->
toCollection(LinkedHashSet<Short>(mapCapacity(size)))\n    }\n}\n\n/**\n * Returns a [Set] of all elements.\n * \n *
The returned set preserves the element iteration order of the original array.\n */
*\npublic fun IntArray.toSet():
Set<Int> {\n    return when (size) {\n        0 -> emptySet()\n        1 -> setOf(this[0])\n        else ->
toCollection(LinkedHashSet<Int>(mapCapacity(size)))\n    }\n}\n\n/**\n * Returns a [Set] of all elements.\n * \n *
The returned set preserves the element iteration order of the original array.\n */
*\npublic fun LongArray.toSet():
Set<Long> {\n    return when (size) {\n        0 -> emptySet()\n        1 -> setOf(this[0])\n        else ->
toCollection(LinkedHashSet<Long>(mapCapacity(size)))\n    }\n}\n\n/**\n * Returns a [Set] of all elements.\n * \n *
The returned set preserves the element iteration order of the original array.\n */
*\npublic fun FloatArray.toSet():
Set<Float> {\n    return when (size) {\n        0 -> emptySet()\n        1 -> setOf(this[0])\n        else ->
toCollection(LinkedHashSet<Float>(mapCapacity(size)))\n    }\n}\n\n/**\n * Returns a [Set] of all elements.\n * \n *
The returned set preserves the element iteration order of the original array.\n */
*\npublic fun DoubleArray.toSet():
Set<Double> {\n    return when (size) {\n        0 -> emptySet()\n        1 -> setOf(this[0])\n        else ->
toCollection(LinkedHashSet<Double>(mapCapacity(size)))\n    }\n}\n\n/**\n * Returns a [Set] of all elements.\n * \n *
The returned set preserves the element iteration order of the original array.\n */
*\npublic fun
BooleanArray.toSet(): Set<Boolean> {\n    return when (size) {\n        0 -> emptySet()\n        1 -> setOf(this[0])\n
else -> toCollection(LinkedHashSet<Boolean>(mapCapacity(size)))\n    }\n}\n\n/**\n * Returns a [Set] of all
elements.\n * \n * The returned set preserves the element iteration order of the original array.\n */
*\npublic fun

```

```

CharArray.toSet(): Set<Char> {\n    return when (size) {\n        0 -> emptySet()\n        1 -> setOf(this[0])\n        else -> toCollection(LinkedHashSet<Char>(mapCapacity(size.coerceAtMost(128))))\n    }\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n *\n * @sample samples.collections.Collections.Transformations.flatMap\n */\npublic inline fun <T, R> Array<out T>.flatMap(transform: (T) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n *\n * @sample samples.collections.Collections.Transformations.flatMap\n */\npublic inline fun <R> ByteArray.flatMap(transform: (Byte) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n *\n * @sample samples.collections.Collections.Transformations.flatMap\n */\npublic inline fun <R> ShortArray.flatMap(transform: (Short) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n *\n * @sample samples.collections.Collections.Transformations.flatMap\n */\npublic inline fun <R> IntArray.flatMap(transform: (Int) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n *\n * @sample samples.collections.Collections.Transformations.flatMap\n */\npublic inline fun <R> LongArray.flatMap(transform: (Long) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n *\n * @sample samples.collections.Collections.Transformations.flatMap\n */\npublic inline fun <R> FloatArray.flatMap(transform: (Float) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n *\n * @sample samples.collections.Collections.Transformations.flatMap\n */\npublic inline fun <R> DoubleArray.flatMap(transform: (Double) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n *\n * @sample samples.collections.Collections.Transformations.flatMap\n */\npublic inline fun <R> BooleanArray.flatMap(transform: (Boolean) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n *\n * @sample samples.collections.Collections.Transformations.flatMap\n */\npublic inline fun <R> CharArray.flatMap(transform: (Char) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n *\n * @sample samples.collections.Collections.Transformations.flatMap\n */\n\n*\/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapSequence")\npublic inline fun <T, R> Array<out T>.flatMap(transform: (T) -> Sequence<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element\n * and its index in the original array.\n *\n * @sample samples.collections.Collections.Transformations.flatMapIndexed\n */\n\n*\/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterable")\n@kotlin.internal.InlineOnly\npublic inline fun <T, R> Array<out T>.flatMapIndexed(transform: (index: Int, T) -> Iterable<R>): List<R> {\n    return flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element\n * and its index in the original array.\n *\n * @sample samples.collections.Collections.Transformations.flatMapIndexed\n */

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterable")\n@kotlin.internal.InlineOnly\npublic
inline fun <R> ByteArray.flatMapIndexed(transform: (index: Int, Byte) -> Iterable<R>): List<R> {\n  return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from
results of [transform] function being invoked on each element\n * and its index in the original array.\n * \n *
@sample samples.collections.Collections.Transformations.flatMapIndexed\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterable")\n@kotlin.internal.InlineOnly\npublic
inline fun <R> ShortArray.flatMapIndexed(transform: (index: Int, Short) -> Iterable<R>): List<R> {\n  return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from
results of [transform] function being invoked on each element\n * and its index in the original array.\n * \n *
@sample samples.collections.Collections.Transformations.flatMapIndexed\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterable")\n@kotlin.internal.InlineOnly\npublic
inline fun <R> IntArray.flatMapIndexed(transform: (index: Int, Int) -> Iterable<R>): List<R> {\n  return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from
results of [transform] function being invoked on each element\n * and its index in the original array.\n * \n *
@sample samples.collections.Collections.Transformations.flatMapIndexed\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterable")\n@kotlin.internal.InlineOnly\npublic
inline fun <R> LongArray.flatMapIndexed(transform: (index: Int, Long) -> Iterable<R>): List<R> {\n  return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from
results of [transform] function being invoked on each element\n * and its index in the original array.\n * \n *
@sample samples.collections.Collections.Transformations.flatMapIndexed\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterable")\n@kotlin.internal.InlineOnly\npublic
inline fun <R> FloatArray.flatMapIndexed(transform: (index: Int, Float) -> Iterable<R>): List<R> {\n  return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from
results of [transform] function being invoked on each element\n * and its index in the original array.\n * \n *
@sample samples.collections.Collections.Transformations.flatMapIndexed\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterable")\n@kotlin.internal.InlineOnly\npublic
inline fun <R> DoubleArray.flatMapIndexed(transform: (index: Int, Double) -> Iterable<R>): List<R> {\n  return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from
results of [transform] function being invoked on each element\n * and its index in the original array.\n * \n *
@sample samples.collections.Collections.Transformations.flatMapIndexed\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterable")\n@kotlin.internal.InlineOnly\npublic
inline fun <R> BooleanArray.flatMapIndexed(transform: (index: Int, Boolean) -> Iterable<R>): List<R> {\n
return flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded
from results of [transform] function being invoked on each element\n * and its index in the original array.\n * \n *
@sample samples.collections.Collections.Transformations.flatMapIndexed\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterable")\n@kotlin.internal.InlineOnly\npublic
inline fun <R> CharArray.flatMapIndexed(transform: (index: Int, Char) -> Iterable<R>): List<R> {\n  return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from
results of [transform] function being invoked on each element\n * and its index in the original array.\n * \n *
@sample samples.collections.Collections.Transformations.flatMapIndexed\n

```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedSequence")\n@kotlin.internal.InlineOnly\npublic inline fun <T, R> Array<out T>.flatMapIndexed(transform: (index: Int, T) -> Sequence<R>): List<R> {\n    return\n    flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Appends all elements yielded from results of\n [transform] function being invoked on each element\n * and its index in the original array, to the given\n [destination].\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterableTo")\n@kotlin.internal.InlineOnly\npublic inline fun <T, R, C : MutableCollection<in R>> Array<out T>.flatMapIndexedTo(destination: C, transform:\n(index: Int, T) -> Iterable<R>): C {\n    var index = 0\n    for (element in this) {\n        val list = transform(index++,\n        element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all elements yielded from\n results of [transform] function being invoked on each element\n * and its index in the original array, to the given\n [destination].\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterableTo")\n@kotlin.internal.InlineOnly\npublic inline fun <R, C : MutableCollection<in R>> ByteArray.flatMapIndexedTo(destination: C, transform: (index: Int,\nByte) -> Iterable<R>): C {\n    var index = 0\n    for (element in this) {\n        val list = transform(index++,\n        element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all elements yielded from\n results of [transform] function being invoked on each element\n * and its index in the original array, to the given\n [destination].\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterableTo")\n@kotlin.internal.InlineOnly\npublic inline fun <R, C : MutableCollection<in R>> ShortArray.flatMapIndexedTo(destination: C, transform: (index: Int,\nShort) -> Iterable<R>): C {\n    var index = 0\n    for (element in this) {\n        val list = transform(index++,\n        element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all elements yielded from\n results of [transform] function being invoked on each element\n * and its index in the original array, to the given\n [destination].\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterableTo")\n@kotlin.internal.InlineOnly\npublic inline fun <R, C : MutableCollection<in R>> IntArray.flatMapIndexedTo(destination: C, transform: (index: Int,\nInt) -> Iterable<R>): C {\n    var index = 0\n    for (element in this) {\n        val list = transform(index++, element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all elements yielded from results of\n [transform] function being invoked on each element\n * and its index in the original array, to the given\n [destination].\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterableTo")\n@kotlin.internal.InlineOnly\npublic inline fun <R, C : MutableCollection<in R>> LongArray.flatMapIndexedTo(destination: C, transform: (index: Int,\nLong) -> Iterable<R>): C {\n    var index = 0\n    for (element in this) {\n        val list = transform(index++,\n        element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all elements yielded from\n results of [transform] function being invoked on each element\n * and its index in the original array, to the given\n [destination].\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterableTo")\n@kotlin.internal.InlineOnly\npublic inline fun <R, C : MutableCollection<in R>> FloatArray.flatMapIndexedTo(destination: C, transform: (index: Int,\nFloat) -> Iterable<R>): C {\n    var index = 0\n    for (element in this) {\n        val list = transform(index++,\n        element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all elements yielded from\n results of [transform] function being invoked on each element\n * and its index in the original array, to the given\n [destination].\n
```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\flatMapIndexedIterableTo")\n@kotlin.internal.InlineOnly\npubli
c inline fun <R, C : MutableCollection<in R>> DoubleArray.flatMapIndexedTo(destination: C, transform: (index:
Int, Double) -> Iterable<R>): C {\n    var index = 0\n    for (element in this) {\n        val list = transform(index++,
element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all elements yielded from
results of [transform] function being invoked on each element\n * and its index in the original array, to the given
[destination].\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\flatMapIndexedIterableTo")\n@kotlin.internal.InlineOnly\npubli
c inline fun <R, C : MutableCollection<in R>> BooleanArray.flatMapIndexedTo(destination: C, transform: (index:
Int, Boolean) -> Iterable<R>): C {\n    var index = 0\n    for (element in this) {\n        val list = transform(index++,
element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all elements yielded from
results of [transform] function being invoked on each element\n * and its index in the original array, to the given
[destination].\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\flatMapIndexedIterableTo")\n@kotlin.internal.InlineOnly\npubli
c inline fun <R, C : MutableCollection<in R>> CharArray.flatMapIndexedTo(destination: C, transform: (index: Int,
Char) -> Iterable<R>): C {\n    var index = 0\n    for (element in this) {\n        val list = transform(index++,
element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all elements yielded from
results of [transform] function being invoked on each element\n * and its index in the original array, to the given
[destination].\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\flatMapIndexedSequenceTo")\n@kotlin.internal.InlineOnly\npubli
c inline fun <T, R, C : MutableCollection<in R>> Array<out T>.flatMapIndexedTo(destination: C, transform:
(index: Int, T) -> Sequence<R>): C {\n    var index = 0\n    for (element in this) {\n        val list =
transform(index++, element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all
elements yielded from results of [transform] function being invoked on each element of original array, to the given
[destination].\n
*\npublic inline fun <T, R, C : MutableCollection<in R>> Array<out T>.flatMapTo(destination: C,
transform: (T) -> Iterable<R>): C {\n    for (element in this) {\n        val list = transform(element)\n
destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all elements yielded from results of
[transform] function being invoked on each element of original array, to the given [destination].\n
*\npublic inline
fun <R, C : MutableCollection<in R>> ByteArray.flatMapTo(destination: C, transform: (Byte) -> Iterable<R>): C
{\n    for (element in this) {\n        val list = transform(element)\n        destination.addAll(list)\n    }\n    return
destination\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each
element of original array, to the given [destination].\n
*\npublic inline fun <R, C : MutableCollection<in R>>
ShortArray.flatMapTo(destination: C, transform: (Short) -> Iterable<R>): C {\n    for (element in this) {\n       
val list = transform(element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all
elements yielded from results of [transform] function being invoked on each element of original array, to the given
[destination].\n
*\npublic inline fun <R, C : MutableCollection<in R>>
IntArray.flatMapTo(destination: C, transform: (Int) -> Iterable<R>): C {\n    for (element in this) {\n       
val list = transform(element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all
elements yielded from results of [transform] function being invoked on each element of original array, to the given
[destination].\n
*\npublic inline
fun <R, C : MutableCollection<in R>> LongArray.flatMapTo(destination: C, transform: (Long) -> Iterable<R>): C
{\n    for (element in this) {\n        val list = transform(element)\n        destination.addAll(list)\n    }\n    return
destination\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each
element of original array, to the given [destination].\n
*\npublic inline fun <R, C : MutableCollection<in R>>
FloatArray.flatMapTo(destination: C, transform: (Float) -> Iterable<R>): C {\n    for (element in this) {\n       
val list = transform(element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all

```

```

elements yielded from results of [transform] function being invoked on each element of original array, to the given
[destination].\n *\npublic inline fun <R, C : MutableCollection<in R>> DoubleArray.flatMapTo(destination: C,
transform: (Double) -> Iterable<R>): C {\n    for (element in this) {\n        val list = transform(element)\n
destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all elements yielded from results of
[transform] function being invoked on each element of original array, to the given [destination].\n *\npublic inline
fun <R, C : MutableCollection<in R>> BooleanArray.flatMapTo(destination: C, transform: (Boolean) ->
Iterable<R>): C {\n    for (element in this) {\n        val list = transform(element)\n        destination.addAll(list)\n
}\n    return destination\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being
invoked on each element of original array, to the given [destination].\n *\npublic inline fun <R, C :
MutableCollection<in R>> CharArray.flatMapTo(destination: C, transform: (Char) -> Iterable<R>): C {\n    for
(element in this) {\n        val list = transform(element)\n        destination.addAll(list)\n    }\n    return
destination\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each
element of original array, to the given [destination].\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapSequenceTo")\npublic inline fun <T, R, C :
MutableCollection<in R>> Array<out T>.flatMapTo(destination: C, transform: (T) -> Sequence<R>): C {\n    for
(element in this) {\n        val list = transform(element)\n        destination.addAll(list)\n    }\n    return
destination\n}\n\n/**\n * Groups elements of the original array by the key returned by the given [keySelector]
function\n * applied to each element and returns a map where each group key is associated with a list of
corresponding elements.\n * \n * The returned map preserves the entry iteration order of the keys produced from the
original array.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n *\npublic inline fun
<T, K> Array<out T>.groupBy(keySelector: (T) -> K): Map<K, List<T>> {\n    return
groupByTo(LinkedHashMap<K, MutableList<T>>(), keySelector)\n}\n\n/**\n * Groups elements of the original
array by the key returned by the given [keySelector] function\n * applied to each element and returns a map where
each group key is associated with a list of corresponding elements.\n * \n * The returned map preserves the entry
iteration order of the keys produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupBy\n *\npublic inline fun <K>
ByteArray.groupBy(keySelector: (Byte) -> K): Map<K, List<Byte>> {\n    return groupByTo(LinkedHashMap<K,
MutableList<Byte>>(), keySelector)\n}\n\n/**\n * Groups elements of the original array by the key returned by the
given [keySelector] function\n * applied to each element and returns a map where each group key is associated with
a list of corresponding elements.\n * \n * The returned map preserves the entry iteration order of the keys produced
from the original array.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n *\npublic
inline fun <K> ShortArray.groupBy(keySelector: (Short) -> K): Map<K, List<Short>> {\n    return
groupByTo(LinkedHashMap<K, MutableList<Short>>(), keySelector)\n}\n\n/**\n * Groups elements of the
original array by the key returned by the given [keySelector] function\n * applied to each element and returns a map
where each group key is associated with a list of corresponding elements.\n * \n * The returned map preserves the
entry iteration order of the keys produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupBy\n *\npublic inline fun <K>
IntArray.groupBy(keySelector: (Int) -> K): Map<K, List<Int>> {\n    return groupByTo(LinkedHashMap<K,
MutableList<Int>>(), keySelector)\n}\n\n/**\n * Groups elements of the original array by the key returned by the
given [keySelector] function\n * applied to each element and returns a map where each group key is associated with
a list of corresponding elements.\n * \n * The returned map preserves the entry iteration order of the keys produced
from the original array.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n *\npublic
inline fun <K> LongArray.groupBy(keySelector: (Long) -> K): Map<K, List<Long>> {\n    return
groupByTo(LinkedHashMap<K, MutableList<Long>>(), keySelector)\n}\n\n/**\n * Groups elements of the
original array by the key returned by the given [keySelector] function\n * applied to each element and returns a map
where each group key is associated with a list of corresponding elements.\n * \n * The returned map preserves the
entry iteration order of the keys produced from the original array.\n * \n * @sample

```

```

samples.collections.Collections.Transformations.groupBy\n *^\npublic inline fun <K>
FloatArray.groupBy(keySelector: (Float) -> K): Map<K, List<Float>> {\n  return groupByTo(LinkedHashMap<K,
MutableList<Float>>(), keySelector)\n}\n\n/**\n * Groups elements of the original array by the key returned by the
given [keySelector] function\n * applied to each element and returns a map where each group key is associated with
a list of corresponding elements.\n * \n * The returned map preserves the entry iteration order of the keys produced
from the original array.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n *^\npublic
inline fun <K> DoubleArray.groupBy(keySelector: (Double) -> K): Map<K, List<Double>> {\n  return
groupByTo(LinkedHashMap<K, MutableList<Double>>(), keySelector)\n}\n\n/**\n * Groups elements of the
original array by the key returned by the given [keySelector] function\n * applied to each element and returns a map
where each group key is associated with a list of corresponding elements.\n * \n * The returned map preserves the
entry iteration order of the keys produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupBy\n *^\npublic inline fun <K>
BooleanArray.groupBy(keySelector: (Boolean) -> K): Map<K, List<Boolean>> {\n  return
groupByTo(LinkedHashMap<K, MutableList<Boolean>>(), keySelector)\n}\n\n/**\n * Groups elements of the
original array by the key returned by the given [keySelector] function\n * applied to each element and returns a map
where each group key is associated with a list of corresponding elements.\n * \n * The returned map preserves the
entry iteration order of the keys produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupBy\n *^\npublic inline fun <K>
CharArray.groupBy(keySelector: (Char) -> K): Map<K, List<Char>> {\n  return groupByTo(LinkedHashMap<K,
MutableList<Char>>(), keySelector)\n}\n\n/**\n * Groups values returned by the [valueTransform] function applied
to each element of the original array\n * by the key returned by the given [keySelector] function applied to the
element\n * and returns a map where each group key is associated with a list of corresponding values.\n * \n * The
returned map preserves the entry iteration order of the keys produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupByKeysAndValues\n *^\npublic inline fun <T, K, V>
Array<out T>.groupBy(keySelector: (T) -> K, valueTransform: (T) -> V): Map<K, List<V>> {\n  return
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups values
returned by the [valueTransform] function applied to each element of the original array\n * by the key returned by
the given [keySelector] function applied to the element\n * and returns a map where each group key is associated
with a list of corresponding values.\n * \n * The returned map preserves the entry iteration order of the keys
produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupByKeysAndValues\n *^\npublic inline fun <K, V>
ByteArray.groupBy(keySelector: (Byte) -> K, valueTransform: (Byte) -> V): Map<K, List<V>> {\n  return
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups values
returned by the [valueTransform] function applied to each element of the original array\n * by the key returned by
the given [keySelector] function applied to the element\n * and returns a map where each group key is associated
with a list of corresponding values.\n * \n * The returned map preserves the entry iteration order of the keys
produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupByKeysAndValues\n *^\npublic inline fun <K, V>
ShortArray.groupBy(keySelector: (Short) -> K, valueTransform: (Short) -> V): Map<K, List<V>> {\n  return
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups values
returned by the [valueTransform] function applied to each element of the original array\n * by the key returned by
the given [keySelector] function applied to the element\n * and returns a map where each group key is associated
with a list of corresponding values.\n * \n * The returned map preserves the entry iteration order of the keys
produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupByKeysAndValues\n *^\npublic inline fun <K, V>
IntArray.groupBy(keySelector: (Int) -> K, valueTransform: (Int) -> V): Map<K, List<V>> {\n  return
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups values
returned by the [valueTransform] function applied to each element of the original array\n * by the key returned by

```

the given [keySelector] function applied to the element\n \* and returns a map where each group key is associated with a list of corresponding values.\n \* \n \* The returned map preserves the entry iteration order of the keys produced from the original array.\n \* \n \* @sample

```
samples.collections.Collections.Transformations.groupByKeyAndValues\n *^\npublic inline fun <K, V>  
LongArray.groupBy(keySelector: (Long) -> K, valueTransform: (Long) -> V): Map<K, List<V>> {\n    return  
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups values  
returned by the [valueTransform] function applied to each element of the original array\n * by the key returned by  
the given [keySelector] function applied to the element\n * and returns a map where each group key is associated  
with a list of corresponding values.\n * \n * The returned map preserves the entry iteration order of the keys  
produced from the original array.\n * \n * @sample
```

```
samples.collections.Collections.Transformations.groupByKeyAndValues\n *^\npublic inline fun <K, V>  
FloatArray.groupBy(keySelector: (Float) -> K, valueTransform: (Float) -> V): Map<K, List<V>> {\n    return  
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups values  
returned by the [valueTransform] function applied to each element of the original array\n * by the key returned by  
the given [keySelector] function applied to the element\n * and returns a map where each group key is associated  
with a list of corresponding values.\n * \n * The returned map preserves the entry iteration order of the keys  
produced from the original array.\n * \n * @sample
```

```
samples.collections.Collections.Transformations.groupByKeyAndValues\n *^\npublic inline fun <K, V>  
DoubleArray.groupBy(keySelector: (Double) -> K, valueTransform: (Double) -> V): Map<K, List<V>> {\n    return  
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups  
values returned by the [valueTransform] function applied to each element of the original array\n * by the key  
returned by the given [keySelector] function applied to the element\n * and returns a map where each group key is  
associated with a list of corresponding values.\n * \n * The returned map preserves the entry iteration order of the  
keys produced from the original array.\n * \n * @sample
```

```
samples.collections.Collections.Transformations.groupByKeyAndValues\n *^\npublic inline fun <K, V>  
BooleanArray.groupBy(keySelector: (Boolean) -> K, valueTransform: (Boolean) -> V): Map<K, List<V>> {\n    return  
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups  
values returned by the [valueTransform] function applied to each element of the original array\n * by the key  
returned by the given [keySelector] function applied to the element\n * and returns a map where each group key is  
associated with a list of corresponding values.\n * \n * The returned map preserves the entry iteration order of the  
keys produced from the original array.\n * \n * @sample
```

```
samples.collections.Collections.Transformations.groupByKeyAndValues\n *^\npublic inline fun <K, V>  
CharArray.groupBy(keySelector: (Char) -> K, valueTransform: (Char) -> V): Map<K, List<V>> {\n    return  
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups elements  
of the original array by the key returned by the given [keySelector] function\n * applied to each element and puts to  
the [destination] map each group key associated with a list of corresponding elements.\n * \n * @return The  
[destination] map.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n *^\npublic inline  
fun <T, K, M : MutableMap<in K, MutableList<T>>> Array<out T>.groupByTo(destination: M, keySelector: (T) -
```

```
> K): M {\n    for (element in this) {\n        val key = keySelector(element)\n        val list = destination.getOrPut(key)  
{ ArrayList<T>() }\n        list.add(element)\n    }\n    return destination\n}\n\n/**\n * Groups elements of the  
original array by the key returned by the given [keySelector] function\n * applied to each element and puts to the  
[destination] map each group key associated with a list of corresponding elements.\n * \n * @return The  
[destination] map.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n *^\npublic inline  
fun <K, M : MutableMap<in K, MutableList<Byte>>> ByteArray.groupByTo(destination: M, keySelector: (Byte) -
```

```

[destination] map.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n *^\npublic inline
fun <K, M : MutableMap<in K, MutableList<Short>>> ShortArray.groupByTo(destination: M, keySelector: (Short)
-> K): M {\n for (element in this) {\n val key = keySelector(element)\n val list =
destination.getOrPut(key) { ArrayList<Short>() }\n list.add(element)\n }\n return destination\n}\n\n/**\n *
Groups elements of the original array by the key returned by the given [keySelector] function\n * applied to each
element and puts to the [destination] map each group key associated with a list of corresponding elements.\n * \n *
@return The [destination] map.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n
*^\npublic inline fun <K, M : MutableMap<in K, MutableList<Int>>> IntArray.groupByTo(destination: M,
keySelector: (Int) -> K): M {\n for (element in this) {\n val key = keySelector(element)\n val list =
destination.getOrPut(key) { ArrayList<Int>() }\n list.add(element)\n }\n return destination\n}\n\n/**\n *
Groups elements of the original array by the key returned by the given [keySelector] function\n * applied to each
element and puts to the [destination] map each group key associated with a list of corresponding elements.\n * \n *
@return The [destination] map.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n
*^\npublic inline fun <K, M : MutableMap<in K, MutableList<Long>>> LongArray.groupByTo(destination: M,
keySelector: (Long) -> K): M {\n for (element in this) {\n val key = keySelector(element)\n val list =
destination.getOrPut(key) { ArrayList<Long>() }\n list.add(element)\n }\n return destination\n}\n\n/**\n *
Groups elements of the original array by the key returned by the given [keySelector] function\n * applied to each
element and puts to the [destination] map each group key associated with a list of corresponding elements.\n * \n *
@return The [destination] map.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n
*^\npublic inline fun <K, M : MutableMap<in K, MutableList<Float>>> FloatArray.groupByTo(destination: M,
keySelector: (Float) -> K): M {\n for (element in this) {\n val key = keySelector(element)\n val list =
destination.getOrPut(key) { ArrayList<Float>() }\n list.add(element)\n }\n return destination\n}\n\n/**\n *
Groups elements of the original array by the key returned by the given [keySelector] function\n * applied to each
element and puts to the [destination] map each group key associated with a list of corresponding elements.\n * \n *
@return The [destination] map.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n
*^\npublic inline fun <K, M : MutableMap<in K, MutableList<Double>>> DoubleArray.groupByTo(destination: M,
keySelector: (Double) -> K): M {\n for (element in this) {\n val key = keySelector(element)\n val list =
destination.getOrPut(key) { ArrayList<Double>() }\n list.add(element)\n }\n return destination\n}\n\n/**\n *
Groups elements of the original array by the key returned by the given [keySelector] function\n * applied to each
element and puts to the [destination] map each group key associated with a list of corresponding elements.\n * \n *
@return The [destination] map.\n * \n * @sample
samples.collections.Collections.Transformations.groupBy\n *^\npublic inline fun <K, M : MutableMap<in K,
MutableList<Boolean>>> BooleanArray.groupByTo(destination:
M, keySelector: (Boolean) -> K): M {\n for (element in this) {\n val key = keySelector(element)\n val list
= destination.getOrPut(key) { ArrayList<Boolean>() }\n list.add(element)\n }\n return
destination\n}\n\n/**\n * Groups elements of the original array by the key returned by the given [keySelector]
function\n * applied to each element and puts to the [destination] map each group key associated with a list of
corresponding elements.\n * \n * @return The [destination] map.\n * \n * @sample
samples.collections.Collections.Transformations.groupBy\n *^\npublic inline fun <K, M : MutableMap<in K,
MutableList<Char>>> CharArray.groupByTo(destination: M, keySelector: (Char) -> K): M {\n for (element in
this) {\n val key = keySelector(element)\n val list = destination.getOrPut(key) { ArrayList<Char>() }\n
list.add(element)\n }\n return destination\n}\n\n/**\n * Groups values returned by the [valueTransform] function
applied to each element of the original array\n * by the key returned by the given [keySelector] function applied to
the element\n * and puts to the [destination] map each group key associated with a list of corresponding values.\n *
\n * @return The [destination] map.\n * \n * @sample
samples.collections.Collections.Transformations.groupByKeysAndValues\n *^\npublic inline fun <T, K, V, M :
MutableMap<in K, MutableList<V>>> Array<out T>.groupByTo(destination: M, keySelector: (T) -> K,
valueTransform: (T) -> V): M {\n for (element in this) {\n val key = keySelector(element)\n val list =
destination.getOrPut(key) { ArrayList<V>() }\n list.add(valueTransform(element))\n }\n return

```

```

destination\n}\n\n/**\n * Groups values returned by the [valueTransform] function applied to each element of the
original array\n * by the key returned by the given [keySelector] function applied to the element\n * and puts to the
[destination] map each group key associated with a list of corresponding values.\n * \n * @return The [destination]
map.\n * \n * @sample samples.collections.Collections.Transformations.groupByKeyAndValues\n */\npublic
inline fun <K, V, M : MutableMap<in K, MutableList<V>>> ByteArray.groupByTo(destination: M, keySelector:
(Byte) -> K, valueTransform: (Byte) -> V): M {\n for (element in this) {\n val key = keySelector(element)\n
val list = destination.getOrPut(key) { ArrayList<V>() }\n list.add(valueTransform(element))\n }\n return
destination\n}\n\n/**\n * Groups values returned by the [valueTransform] function applied to each element of the
original array\n * by the key returned by the given [keySelector] function applied to the element\n * and puts to the
[destination] map each group key associated with a list of corresponding values.\n * \n * @return The [destination]
map.\n * \n * @sample samples.collections.Collections.Transformations.groupByKeyAndValues\n */\npublic
inline fun <K, V, M : MutableMap<in K, MutableList<V>>> ShortArray.groupByTo(destination: M, keySelector:
(Short) -> K, valueTransform: (Short) -> V): M {\n for (element in this) {\n val key = keySelector(element)\n
val list = destination.getOrPut(key) { ArrayList<V>() }\n list.add(valueTransform(element))\n }\n return
destination\n}\n\n/**\n * Groups values returned by the [valueTransform] function applied to each element of the
original array\n * by the key returned by the given [keySelector] function applied to the element\n * and puts to the
[destination] map each group key associated with a list of corresponding values.\n * \n * @return The [destination]
map.\n * \n * @sample samples.collections.Collections.Transformations.groupByKeyAndValues\n */\npublic
inline fun <K, V, M : MutableMap<in K, MutableList<V>>> IntArray.groupByTo(destination: M, keySelector:
(Int) -> K, valueTransform: (Int) -> V): M {\n for (element in this) {\n val key = keySelector(element)\n
val list = destination.getOrPut(key) { ArrayList<V>() }\n list.add(valueTransform(element))\n }\n return
destination\n}\n\n/**\n * Groups values returned by the [valueTransform] function applied to each element of the
original array\n * by the key returned by the given [keySelector] function applied to the element\n * and puts to the
[destination] map each group key associated with a list of corresponding values.\n * \n * @return The [destination]
map.\n * \n * @sample samples.collections.Collections.Transformations.groupByKeyAndValues\n */\npublic
inline fun <K, V, M : MutableMap<in K, MutableList<V>>> LongArray.groupByTo(destination: M, keySelector:
(Long) -> K, valueTransform: (Long) -> V): M {\n for (element in this) {\n val key = keySelector(element)\n
val list = destination.getOrPut(key) { ArrayList<V>() }\n list.add(valueTransform(element))\n }\n return
destination\n}\n\n/**\n * Groups values returned by the [valueTransform] function applied to each element of the
original array\n * by the key returned by the given [keySelector] function applied to the element\n * and puts to the
[destination] map each group key associated with a list of corresponding values.\n * \n * @return The [destination]
map.\n * \n * @sample samples.collections.Collections.Transformations.groupByKeyAndValues\n */\npublic
inline fun <K, V, M : MutableMap<in K, MutableList<V>>> FloatArray.groupByTo(destination: M, keySelector:
(Float) -> K, valueTransform: (Float) -> V): M {\n for (element in this) {\n val key = keySelector(element)\n
val list = destination.getOrPut(key) { ArrayList<V>() }\n list.add(valueTransform(element))\n }\n return
destination\n}\n\n/**\n * Groups values returned by the [valueTransform] function applied to each element of the
original array\n * by the key returned by the given [keySelector] function applied to the element\n * and puts to the
[destination] map each group key associated with a list of corresponding values.\n * \n * @return The [destination]
map.\n * \n * @sample samples.collections.Collections.Transformations.groupByKeyAndValues\n */\npublic
inline fun <K, V, M : MutableMap<in K, MutableList<V>>> DoubleArray.groupByTo(destination: M, keySelector:
(Double) -> K, valueTransform: (Double) -> V): M {\n for (element in this) {\n val key =
keySelector(element)\n val list = destination.getOrPut(key) { ArrayList<V>() }\n
list.add(valueTransform(element))\n }\n return destination\n}\n\n/**\n * Groups values returned by the
[valueTransform] function applied to each element of the original array\n * by the key returned by the given
[keySelector] function applied to the element\n * and puts to the [destination] map each group key associated with a
list of corresponding values.\n * \n * @return The [destination] map.\n * \n * @sample
samples.collections.Collections.Transformations.groupByKeyAndValues\n */\npublic inline fun <K, V, M :
MutableMap<in K, MutableList<V>>> BooleanArray.groupByTo(destination: M, keySelector: (Boolean) -> K,

```

```

valueTransform: (Boolean) -> V): M { \n for (element in this) { \n val key = keySelector(element)\n val list
= destination.getOrPut(key) { ArrayList<V>() }\n list.add(valueTransform(element))\n }\n return
destination\n}\n\n/**\n * Groups values returned by the [valueTransform] function applied to each element of the
original array\n * by the key returned by the given [keySelector] function applied to the element\n * and puts to the
[destination] map each group key associated with a list of corresponding values.\n * \n * @return The [destination]
map.\n * \n * @sample samples.collections.Collections.Transformations.groupByKeyAndValues\n */\npublic
inline fun <K, V, M : MutableMap<in K, MutableList<V>>> CharArray.groupByTo(destination: M, keySelector:
(Char) -> K, valueTransform: (Char) -> V): M { \n for (element in this) { \n val key = keySelector(element)\n
val list = destination.getOrPut(key) { ArrayList<V>() }\n list.add(valueTransform(element))\n }\n return
destination\n}\n\n/**\n * Creates a [Grouping] source from an array to be used later with one of group-and-fold
operations\n * using the specified [keySelector] function to extract a key from each element.\n * \n * @sample
samples.collections.Grouping.groupingByEachCount\n */\n@SinceKotlin("1.1")\npublic inline fun <T, K>
Array<out T>.groupingBy(crossinline keySelector: (T) -> K): Grouping<T, K> { \n return object : Grouping<T,
K> { \n override fun sourceIterator(): Iterator<T> = this@groupingBy.iterator()\n override fun
keyOf(element: T): K = keySelector(element)\n }\n}\n\n/**\n * Returns a list containing the results of applying
the given [transform] function\n * to each element in the original array.\n * \n * @sample
samples.collections.Collections.Transformations.map\n */\npublic inline fun <T, R> Array<out T>.map(transform:
(T) -> R): List<R> { \n return mapTo(ArrayList<R>(size), transform)\n}\n\n/**\n * Returns a list containing the
results of applying the given [transform] function\n * to each element in the original array.\n * \n * @sample
samples.collections.Collections.Transformations.map\n */\npublic inline fun <R> ByteArray.map(transform: (Byte)
-> R): List<R> { \n return mapTo(ArrayList<R>(size), transform)\n}\n\n/**\n * Returns a list containing the
results of applying the given [transform] function\n * to each element in the original array.\n * \n * @sample
samples.collections.Collections.Transformations.map\n */\npublic inline fun <R> ShortArray.map(transform:
(Short) -> R): List<R> { \n return mapTo(ArrayList<R>(size), transform)\n}\n\n/**\n * Returns a list containing
the results of applying the given [transform] function\n * to each element in the original array.\n * \n * @sample
samples.collections.Collections.Transformations.map\n */\npublic inline fun <R> IntArray.map(transform: (Int) ->
R): List<R> { \n return mapTo(ArrayList<R>(size), transform)\n}\n\n/**\n * Returns a list containing the results
of applying the given [transform] function\n * to each element in the original array.\n * \n * @sample
samples.collections.Collections.Transformations.map\n */\npublic inline fun <R> LongArray.map(transform:
(Long) -> R): List<R> { \n return mapTo(ArrayList<R>(size), transform)\n}\n\n/**\n * Returns a list containing
the results of applying the given [transform] function\n * to each element in the original array.\n * \n * @sample
samples.collections.Collections.Transformations.map\n */\npublic inline fun <R> FloatArray.map(transform: (Float)
-> R): List<R> { \n return mapTo(ArrayList<R>(size), transform)\n}\n\n/**\n * Returns a list containing the
results of applying the given [transform] function\n * to each element in the original array.\n * \n * @sample
samples.collections.Collections.Transformations.map\n */\npublic inline fun <R> DoubleArray.map(transform:
(Double) -> R): List<R> { \n return mapTo(ArrayList<R>(size), transform)\n}\n\n/**\n * Returns a list containing
the results of applying the given [transform] function\n * to each element in the original array.\n * \n * @sample
samples.collections.Collections.Transformations.map\n */\npublic inline fun <R> BooleanArray.map(transform:
(Boolean) -> R): List<R> { \n return mapTo(ArrayList<R>(size), transform)\n}\n\n/**\n * Returns a list
containing the results of applying the given [transform] function\n * to each element in the original array.\n * \n *
@sample samples.collections.Collections.Transformations.map\n */\npublic inline fun <R>
CharArray.map(transform: (Char) -> R): List<R> { \n return mapTo(ArrayList<R>(size), transform)\n}\n\n/**\n *
Returns a list containing the results of applying the given [transform] function\n * to each element and its index in
the original array.\n * @param [transform] function that takes the index of an element and the element itself\n * and
returns the result of the transform applied to the element.\n */\npublic inline fun <T, R> Array<out
T>.mapIndexed(transform: (index: Int, T) -> R): List<R> { \n return mapIndexedTo(ArrayList<R>(size),
transform)\n}\n\n/**\n * Returns a list containing the results of applying the given [transform] function\n * to each
element and its index in the original array.\n * @param [transform] function that takes the index of an element and

```

the element itself and returns the result of the transform applied to the element.

```

public inline fun <R>
ByteArray.mapIndexed(transform: (index: Int, Byte) -> R): List<R> {
    return
    mapIndexedTo(ArrayList<R>(size), transform)
}

```

Returns a list containing the results of applying the given [transform] function to each element and its index in the original array. @param [transform] function that takes the index of an element and the element itself and returns the result of the transform applied to the element.

```

public inline fun <R> ShortArray.mapIndexed(transform: (index: Int, Short) -> R): List<R> {
    return mapIndexedTo(ArrayList<R>(size), transform)
}

```

Returns a list containing the results of applying the given [transform] function to each element and its index in the original array. @param [transform] function that takes the index of an element and the element itself and returns the result of the transform applied to the element.

```

public inline fun <R> IntArray.mapIndexed(transform: (index: Int, Int) -> R): List<R> {
    return mapIndexedTo(ArrayList<R>(size), transform)
}

```

Returns a list containing the results of applying the given [transform] function to each element and its index in the original array. @param [transform] function that takes the index of an element and the element itself and returns the result of the transform applied to the element.

```

public inline fun <R> LongArray.mapIndexed(transform: (index: Int, Long) -> R): List<R> {
    return mapIndexedTo(ArrayList<R>(size), transform)
}

```

Returns a list containing the results of applying the given [transform] function to each element and its index in the original array. @param [transform] function that takes the index of an element and the element itself and returns the result of the transform applied to the element.

```

public inline fun <R> FloatArray.mapIndexed(transform: (index: Int, Float) -> R): List<R> {
    return mapIndexedTo(ArrayList<R>(size), transform)
}

```

Returns a list containing the results of applying the given [transform] function to each element and its index in the original array. @param [transform] function that takes the index of an element and the element itself and returns the result of the transform applied to the element.

```

public inline fun <R> DoubleArray.mapIndexed(transform: (index: Int, Double) -> R): List<R> {
    return mapIndexedTo(ArrayList<R>(size), transform)
}

```

Returns a list containing the results of applying the given [transform] function to each element and its index in the original array. @param [transform] function that takes the index of an element and the element itself and returns the result of the transform applied to the element.

```

public inline fun <R> BooleanArray.mapIndexed(transform: (index: Int, Boolean) -> R): List<R> {
    return mapIndexedTo(ArrayList<R>(size), transform)
}

```

Returns a list containing the results of applying the given [transform] function to each element and its index in the original array. @param [transform] function that takes the index of an element and the element itself and returns the result of the transform applied to the element.

```

public inline fun <R> CharArray.mapIndexed(transform: (index: Int, Char) -> R): List<R> {
    return
    mapIndexedTo(ArrayList<R>(size), transform)
}

```

Returns a list containing only the non-null results of applying the given [transform] function to each element and its index in the original array. @param [transform] function that takes the index of an element and the element itself and returns the result of the transform applied to the element.

```

public inline fun <T, R : Any> Array<out T>.mapIndexedNotNull(transform: (index: Int, T) -> R?): List<R> {
    return
    mapIndexedNotNullTo(ArrayList<R>(), transform)
}

```

Applies the given [transform] function to each element and its index in the original array and appends only the non-null results to the given [destination]. @param [transform] function that takes the index of an element and the element itself and returns the result of the transform applied to the element.

```

public inline fun <T, R : Any, C : MutableCollection<in R>> Array<out T>.mapIndexedNotNullTo(destination: C, transform: (index: Int, T) -> R?): C {
    forEachIndexed { index, element -> transform(index, element)?.let { destination.add(it) } }
    return destination
}

```

Applies the given [transform] function to each element and its index in the original array and appends the results to the given [destination]. @param [transform] function that takes the index of an element and the element itself and returns the result of the transform applied to the element.

```

public inline fun <T, R, C : MutableCollection<in R>> Array<out T>.mapIndexedTo(destination: C, transform: (index: Int, T) -> R): C {
    var index = 0
    for (item in this)
        destination.add(transform(index++, item))
    return destination
}

```

Applies the given [transform] function to each element and its index in the original array and appends the results to the given

```

[destination].\n * @param [transform] function that takes the index of an element and the element itself\n * and
returns the result of the transform applied to the element.\n *\npublic inline fun <R, C : MutableCollection<in R>>
ByteArray.mapIndexedTo(destination: C, transform: (index: Int, Byte) -> R): C {\n    var index = 0\n    for (item in
this)\n        destination.add(transform(index++, item))\n    return destination\n}\n\n/**\n * Applies the given
[transform] function to each element and its index in the original array\n * and appends the results to the given
[destination].\n * @param [transform] function that takes the index of an element and the element itself\n * and
returns the result of the transform applied to the element.\n *\npublic inline fun <R, C : MutableCollection<in R>>
ShortArray.mapIndexedTo(destination: C, transform: (index: Int, Short) -> R): C {\n    var index = 0\n    for (item in
this)\n        destination.add(transform(index++, item))\n    return destination\n}\n\n/**\n * Applies the given
[transform] function to each element and its index in the original array\n * and appends the results to the given
[destination].\n * @param [transform] function that takes the index of an element and the element itself\n * and
returns the result of the transform applied to the element.\n *\npublic inline fun <R, C : MutableCollection<in R>>
IntArray.mapIndexedTo(destination: C, transform: (index: Int, Int) -> R): C {\n    var index = 0\n    for (item in
this)\n        destination.add(transform(index++, item))\n    return destination\n}\n\n/**\n * Applies the given
[transform] function to each element and its index in the original array\n * and appends the results to the given
[destination].\n * @param [transform] function that takes the index of an element and the element itself\n * and
returns the result of the transform applied to the element.\n *\npublic inline fun <R, C : MutableCollection<in R>>
LongArray.mapIndexedTo(destination: C, transform: (index: Int, Long) -> R): C {\n    var index = 0\n    for (item in
this)\n        destination.add(transform(index++, item))\n    return destination\n}\n\n/**\n * Applies the given
[transform] function to each element and its index in the original array\n * and appends the results to the given
[destination].\n * @param [transform] function that takes the index of an element and the element itself\n * and
returns the result of the transform applied to the element.\n *\npublic inline fun <R, C : MutableCollection<in R>>
FloatArray.mapIndexedTo(destination: C, transform: (index: Int, Float) -> R): C {\n    var index = 0\n    for (item in
this)\n        destination.add(transform(index++, item))\n    return destination\n}\n\n/**\n * Applies the given
[transform] function to each element and its index in the original array\n * and appends the results to the given
[destination].\n * @param [transform] function that takes the index of an element and the element itself\n * and
returns the result of the transform applied to the element.\n *\npublic inline fun <R, C : MutableCollection<in R>>
DoubleArray.mapIndexedTo(destination: C, transform: (index: Int, Double) -> R): C {\n    var index = 0\n    for
(item in this)\n        destination.add(transform(index++, item))\n    return destination\n}\n\n/**\n * Applies the given
[transform] function to each element and its index in the original array\n * and appends the results to the given
[destination].\n * @param [transform] function that takes the index of an element and the element itself\n * and
returns the result of the transform applied to the element.\n *\npublic inline fun <R, C : MutableCollection<in R>>
BooleanArray.mapIndexedTo(destination: C, transform: (index: Int, Boolean) -> R): C {\n    var index = 0\n    for
(item in this)\n        destination.add(transform(index++, item))\n    return destination\n}\n\n/**\n * Applies the given
[transform] function to each element and its index in the original array\n * and appends the results to the given
[destination].\n * @param [transform] function that takes the index of an element and the element itself\n * and
returns the result of the transform applied to the element.\n *\npublic inline fun <R, C : MutableCollection<in R>>
CharArray.mapIndexedTo(destination: C, transform: (index: Int, Char) -> R): C {\n    var index = 0\n    for (item in
this)\n        destination.add(transform(index++, item))\n    return destination\n}\n\n/**\n * Returns a list containing
only the non-null results of applying the given [transform] function\n * to each element in the original array.\n * \n *
@sample samples.collections.Collections.Transformations.mapNotNull\n *\npublic inline fun <T, R : Any>
Array<out T>.mapNotNull(transform: (T) -> R?): List<R> {\n    return mapNotNullTo(ArrayList<R>(),
transform)\n}\n\n/**\n * Applies the given [transform] function to each element in the original array\n * and
appends only the non-null results to the given [destination].\n *\npublic inline fun <T, R : Any, C :
MutableCollection<in R>> Array<out T>.mapNotNullTo(destination: C, transform: (T) -> R?): C {\n    forEach {
element -> transform(element)?.let { destination.add(it) } }\n    return destination\n}\n\n/**\n * Applies the given
[transform] function to each element of the original array\n * and appends the results to the given [destination].\n
*\npublic inline fun <T, R, C : MutableCollection<in R>> Array<out T>.mapTo(destination: C, transform: (T) ->

```

```

R): C {\n  for (item in this)\n    destination.add(transform(item))\n  return destination}\n\n/**\n * Applies the
given [transform] function to each element of the original array\n * and appends the results to the given
[destination].\n */\npublic inline fun <R, C : MutableCollection<in R>> ByteArray.mapTo(destination: C,
transform: (Byte) -> R): C {\n  for (item in this)\n    destination.add(transform(item))\n  return
destination}\n\n/**\n * Applies the given [transform] function to each element of the original array\n * and
appends the results to the given [destination].\n */\npublic inline fun <R, C : MutableCollection<in R>>
ShortArray.mapTo(destination: C, transform: (Short) -> R): C {\n  for (item in this)\n
destination.add(transform(item))\n  return destination}\n\n/**\n * Applies the given [transform] function to each
element of the original array\n * and appends the results to the given [destination].\n */\npublic inline fun <R, C :
MutableCollection<in R>> IntArray.mapTo(destination: C, transform: (Int) -> R): C {\n  for (item in this)\n
destination.add(transform(item))\n  return destination}\n\n/**\n * Applies the given [transform] function to each
element of the original array\n * and appends the results to the given [destination].\n */\npublic inline fun <R, C :
MutableCollection<in R>> LongArray.mapTo(destination: C, transform: (Long) -> R): C {\n  for (item in this)\n
destination.add(transform(item))\n  return destination}\n\n/**\n * Applies the given [transform] function to
each element of the original array\n * and appends the results to the given [destination].\n */\npublic inline fun <R,
C : MutableCollection<in R>> FloatArray.mapTo(destination: C, transform: (Float) -> R): C {\n  for (item in
this)\n  destination.add(transform(item))\n  return destination}\n\n/**\n * Applies the given [transform]
function to each element of the original array\n * and appends the results to the given [destination].\n */\npublic
inline fun <R, C : MutableCollection<in R>> DoubleArray.mapTo(destination: C, transform: (Double) -> R): C {\n
for (item in this)\n  destination.add(transform(item))\n  return destination}\n\n/**\n * Applies the given
[transform] function to each element of the original array\n * and appends the results to the given [destination].\n
*/\npublic inline fun <R, C : MutableCollection<in R>> BooleanArray.mapTo(destination: C, transform: (Boolean)
-> R): C {\n  for (item in this)\n  destination.add(transform(item))\n  return destination}\n\n/**\n * Applies
the given [transform] function to each element of the original array\n * and appends the results to the given
[destination].\n */\npublic inline fun <R, C : MutableCollection<in R>> CharArray.mapTo(destination: C,
transform: (Char) -> R): C {\n  for (item in this)\n  destination.add(transform(item))\n  return
destination}\n\n/**\n * Returns a lazy [Iterable] that wraps each element of the original array\n * into an
[IndexValue] containing the index of that element and the element itself.\n */\npublic fun <T> Array<out
T>.withIndex(): Iterable<IndexedValue<T>> {\n  return IndexingIterable { iterator() }\n}\n\n/**\n * Returns a
lazy [Iterable] that wraps each element of the original array\n * into an [IndexedValue] containing the index of that
element and the element itself.\n */\npublic fun ByteArray.withIndex(): Iterable<IndexedValue<Byte>> {\n  return
IndexingIterable { iterator() }\n}\n\n/**\n * Returns a lazy [Iterable] that wraps each element of the original array\n
* into an [IndexedValue] containing the index of that element and the element itself.\n */\npublic fun
ShortArray.withIndex(): Iterable<IndexedValue<Short>> {\n  return IndexingIterable { iterator() }\n}\n\n/**\n *
Returns a lazy [Iterable] that wraps each element of the original array\n * into an [IndexedValue] containing the
index of that element and the element itself.\n */\npublic fun IntArray.withIndex(): Iterable<IndexedValue<Int>>
{\n  return IndexingIterable { iterator() }\n}\n\n/**\n * Returns a lazy [Iterable] that wraps each element of the
original array\n * into an [IndexedValue] containing the index of that element and the element itself.\n */\npublic
fun LongArray.withIndex(): Iterable<IndexedValue<Long>> {\n  return IndexingIterable { iterator() }\n}\n\n/**\n
* Returns a lazy [Iterable] that wraps each element of the original array\n * into an [IndexedValue] containing the
index of that element and the element itself.\n */\npublic fun FloatArray.withIndex():
Iterable<IndexedValue<Float>> {\n  return IndexingIterable { iterator() }\n}\n\n/**\n * Returns a lazy [Iterable]
that wraps each element of the original array\n * into an [IndexedValue] containing the index of that element and the
element itself.\n */\npublic fun DoubleArray.withIndex(): Iterable<IndexedValue<Double>> {\n  return
IndexingIterable { iterator() }\n}\n\n/**\n * Returns a lazy [Iterable] that wraps each element of the original array\n
* into an [IndexedValue] containing the index of that element and the element itself.\n */\npublic fun
BooleanArray.withIndex(): Iterable<IndexedValue<Boolean>> {\n  return IndexingIterable { iterator()
}\n}\n\n/**\n * Returns a lazy [Iterable] that wraps each element of the original array\n * into an [IndexedValue]

```

containing the index of that element and the element itself.

```

public fun CharArray.withIndex():
Iterable<IndexedValue<Char>> {
    return IndexingIterable { iterator() }
}

```

Returns a list containing only distinct elements from the given array. Among equal elements of the given array, only the first one will be present in the resulting list. The elements in the resulting list are in the same order as they were in the source array.

```

@sample samples.collections.Collections.Transformations.distinctAndDistinctBy

```

public fun <T> Array<out T>.distinct(): List<T> {

```

    return this.toMutableSet().toList()
}

```

Returns a list containing only distinct elements from the given array. The elements in the resulting list are in the same order as they were in the source array.

```

@sample
samples.collections.Collections.Transformations.distinctAndDistinctBy

```

public fun ByteArray.distinct(): List<Byte> {

```

    return this.toMutableSet().toList()
}

```

Returns a list containing only distinct elements from the given array. The elements in the resulting list are in the same order as they were in the source array.

```

@sample samples.collections.Collections.Transformations.distinctAndDistinctBy

```

public fun ShortArray.distinct(): List<Short> {

```

    return this.toMutableSet().toList()
}

```

Returns a list containing only distinct elements from the given array. The elements in the resulting list are in the same order as they were in the source array.

```

@sample
samples.collections.Collections.Transformations.distinctAndDistinctBy

```

public fun IntArray.distinct(): List<Int> {

```

    return this.toMutableSet().toList()
}

```

Returns a list containing only distinct elements from the given array. The elements in the resulting list are in the same order as they were in the source array.

```

@sample samples.collections.Collections.Transformations.distinctAndDistinctBy

```

public fun LongArray.distinct(): List<Long> {

```

    return this.toMutableSet().toList()
}

```

Returns a list containing only distinct elements from the given array. The elements in the resulting list are in the same order as they were in the source array.

```

@sample
samples.collections.Collections.Transformations.distinctAndDistinctBy

```

public fun FloatArray.distinct(): List<Float> {

```

    return this.toMutableSet().toList()
}

```

Returns a list containing only distinct elements from the given array. The elements in the resulting list are in the same order as they were in the source array.

```

@sample samples.collections.Collections.Transformations.distinctAndDistinctBy

```

public fun DoubleArray.distinct(): List<Double> {

```

    return this.toMutableSet().toList()
}

```

Returns a list containing only distinct elements from the given array. The elements in the resulting list are in the same order as they were in the source array.

```

@sample
samples.collections.Collections.Transformations.distinctAndDistinctBy

```

public fun BooleanArray.distinct(): List<Boolean> {

```

    return this.toMutableSet().toList()
}

```

Returns a list containing only distinct elements from the given array. The elements in the resulting list are in the same order as they were in the source array.

```

@sample samples.collections.Collections.Transformations.distinctAndDistinctBy

```

public fun CharArray.distinct(): List<Char> {

```

    return this.toMutableSet().toList()
}

```

Returns a list containing only elements from the given array having distinct keys returned by the given [selector] function. Among elements of the given array with equal keys, only the first one will be present in the resulting list. The elements in the resulting list are in the same order as they were in the source array.

```

@sample
samples.collections.Collections.Transformations.distinctAndDistinctBy

```

public inline fun <T, K> Array<out T>.distinctBy(selector: (T) -> K): List<T> {

```

    val set = HashSet<K>()
    val list = ArrayList<T>()
    for (e in this) {
        val key = selector(e)
        if (set.add(key))
            list.add(e)
    }
    return list
}

```

Returns a list containing only elements from the given array having distinct keys returned by the given [selector] function. The elements in the resulting list are in the same order as they were in the source array.

```

@sample samples.collections.Collections.Transformations.distinctAndDistinctBy

```

public inline fun <K> ByteArray.distinctBy(selector: (Byte) -> K): List<Byte> {

```

    val set = HashSet<K>()
    val list = ArrayList<Byte>()
    for (e in this) {
        val key = selector(e)
        if (set.add(key))
            list.add(e)
    }
    return list
}

```

Returns a list containing only elements from the given array having distinct keys returned by the given [selector] function. The elements in the resulting list are in the same order as they were in the source array.

```

@sample

```

```

samples.collections.Collections.Transformations.distinctAndDistinctBy\n *^\\npublic inline fun <K>
ShortArray.distinctBy(selector: (Short) -> K): List<Short> {\n    val set = HashSet<K>()\n    val list =
ArrayList<Short>()\n    for (e in this) {\n        val key = selector(e)\n        if (set.add(key))\n            list.add(e)\n    }\n    return list\n}\n\n/**\n * Returns a list containing only elements from the given array\n * having distinct keys
returned by the given [selector] function.\n * \n * The elements in the resulting list are in the same order as they
were in the source array.\n * \n * @sample
samples.collections.Collections.Transformations.distinctAndDistinctBy\n *^\\npublic inline fun <K>
IntArray.distinctBy(selector: (Int) -> K): List<Int> {\n    val set = HashSet<K>()\n    val list = ArrayList<Int>()\n
for (e in this) {\n        val key = selector(e)\n        if (set.add(key))\n            list.add(e)\n    }\n    return list\n}\n\n/**\n * Returns a list containing only elements from the given array\n * having distinct keys returned by the given
[selector] function.\n * \n * The elements in the resulting list are in the same order as they were in the source
array.\n * \n * @sample
samples.collections.Collections.Transformations.distinctAndDistinctBy\n *^\\npublic inline
fun <K> LongArray.distinctBy(selector: (Long) -> K): List<Long> {\n    val set = HashSet<K>()\n    val list =
ArrayList<Long>()\n    for (e in this) {\n        val key = selector(e)\n        if (set.add(key))\n            list.add(e)\n    }\n    return list\n}\n\n/**\n * Returns a list containing only elements from the given array\n * having distinct keys
returned by the given [selector] function.\n * \n * The elements in the resulting list are in the same order as they
were in the source array.\n * \n * @sample
samples.collections.Collections.Transformations.distinctAndDistinctBy\n *^\\npublic inline fun <K>
FloatArray.distinctBy(selector: (Float) -> K): List<Float> {\n    val set = HashSet<K>()\n    val list =
ArrayList<Float>()\n    for (e in this) {\n        val key = selector(e)\n        if (set.add(key))\n            list.add(e)\n    }\n    return list\n}\n\n/**\n * Returns a list containing only elements from the given array\n * having distinct keys
returned by the given [selector] function.\n * \n * The elements in the resulting list are in the same order as they
were in the source array.\n * \n * @sample
samples.collections.Collections.Transformations.distinctAndDistinctBy\n *^\\npublic inline fun <K>
DoubleArray.distinctBy(selector: (Double) -> K): List<Double> {\n    val set = HashSet<K>()\n    val list =
ArrayList<Double>()\n    for (e in this) {\n        val key = selector(e)\n        if (set.add(key))\n            list.add(e)\n    }\n    return list\n}\n\n/**\n * Returns a list containing only elements from the given array\n * having distinct keys
returned by the given [selector] function.\n * \n * The elements in the resulting list are in the same order as they
were in the source array.\n * \n * @sample
samples.collections.Collections.Transformations.distinctAndDistinctBy\n *^\\npublic inline fun <K>
BooleanArray.distinctBy(selector: (Boolean) -> K): List<Boolean> {\n    val set = HashSet<K>()\n    val list =
ArrayList<Boolean>()\n    for (e in this) {\n        val key = selector(e)\n        if (set.add(key))\n            list.add(e)\n    }\n    return list\n}\n\n/**\n * Returns a list containing only elements from the given array\n * having distinct keys
returned by the given [selector] function.\n * \n * The elements in the resulting list are in the same order as they
were in the source array.\n * \n * @sample
samples.collections.Collections.Transformations.distinctAndDistinctBy\n *^\\npublic inline fun <K>
CharArray.distinctBy(selector: (Char) -> K): List<Char> {\n    val set = HashSet<K>()\n    val list =
ArrayList<Char>()\n    for (e in this) {\n        val key = selector(e)\n        if (set.add(key))\n            list.add(e)\n    }\n    return list\n}\n\n/**\n * Returns a set containing all elements that are contained by both this array and the specified
collection.\n * \n * The returned set preserves the element iteration order of the original array.\n * \n * To get a set
containing all elements that are contained at least in one of these collections use [union].\n *^\\npublic infix fun <T>
Array<out T>.intersect(other: Iterable<T>): Set<T> {\n    val set = this.toMutableSet()\n    set.retainAll(other)\n    return set\n}\n\n/**\n * Returns a set containing all elements that are contained by both this array and the specified
collection.\n * \n * The returned set preserves the element iteration order of the original array.\n * \n * To get a set
containing all elements that are contained at least in one of these collections use [union].\n *^\\npublic infix fun
ByteArray.intersect(other: Iterable<Byte>): Set<Byte> {\n    val set = this.toMutableSet()\n    set.retainAll(other)\n    return set\n}\n\n/**\n * Returns a set containing all elements that are contained by both this array and the specified
collection.\n * \n * The returned set preserves the element iteration order of the original array.\n * \n * To get a set

```

containing all elements that are contained at least in one of these collections use [union].

```

public infix fun ShortArray.intersect(other: Iterable<Short>): Set<Short> {
    val set = this.toMutableSet()
    set.retainAll(other)
}

```

Returns a set containing all elements that are contained by both this array and the specified collection. The returned set preserves the element iteration order of the original array.

To get a set containing all elements that are contained at least in one of these collections use [union].

```

public infix fun IntArray.intersect(other: Iterable<Int>): Set<Int> {
    val set = this.toMutableSet()
    set.retainAll(other)
}

```

Returns a set containing all elements that are contained by both this array and the specified collection. The returned set preserves the element iteration order of the original array.

To get a set containing all elements that are contained at least in one of these collections use [union].

```

public infix fun LongArray.intersect(other: Iterable<Long>): Set<Long> {
    val set = this.toMutableSet()
    set.retainAll(other)
}

```

Returns a set containing all elements that are contained by both this array and the specified collection. The returned set preserves the element iteration order of the original array.

To get a set containing all elements that are contained at least in one of these collections use [union].

```

public infix fun FloatArray.intersect(other: Iterable<Float>): Set<Float> {
    val set = this.toMutableSet()
    set.retainAll(other)
}

```

Returns a set containing all elements that are contained by both this array and the specified collection. The returned set preserves the element iteration order of the original array.

To get a set containing all elements that are contained at least in one of these collections use [union].

```

public infix fun DoubleArray.intersect(other: Iterable<Double>): Set<Double> {
    val set = this.toMutableSet()
    set.retainAll(other)
}

```

Returns a set containing all elements that are contained by both this array and the specified collection. The returned set preserves the element iteration order of the original array.

To get a set containing all elements that are contained at least in one of these collections use [union].

```

public infix fun BooleanArray.intersect(other: Iterable<Boolean>): Set<Boolean> {
    val set = this.toMutableSet()
    set.retainAll(other)
}

```

Returns a set containing all elements that are contained by both this array and the specified collection. The returned set preserves the element iteration order of the original array.

To get a set containing all elements that are contained at least in one of these collections use [union].

```

public infix fun CharArray.intersect(other: Iterable<Char>): Set<Char> {
    val set = this.toMutableSet()
    set.retainAll(other)
}

```

Returns a set containing all elements that are contained by this array and not contained by the specified collection. The returned set preserves the element iteration order of the original array.

```

public infix fun <T> Array<out T>.subtract(other: Iterable<T>): Set<T> {
    val set = this.toMutableSet()
    set.removeAll(other)
}

```

Returns a set containing all elements that are contained by this array and not contained by the specified collection. The returned set preserves the element iteration order of the original array.

```

public infix fun ByteArray.subtract(other: Iterable<Byte>): Set<Byte> {
    val set = this.toMutableSet()
    set.removeAll(other)
}

```

Returns a set containing all elements that are contained by this array and not contained by the specified collection. The returned set preserves the element iteration order of the original array.

```

public infix fun ShortArray.subtract(other: Iterable<Short>): Set<Short> {
    val set = this.toMutableSet()
    set.removeAll(other)
}

```

Returns a set containing all elements that are contained by this array and not contained by the specified collection. The returned set preserves the element iteration order of the original array.

```

public infix fun IntArray.subtract(other: Iterable<Int>): Set<Int> {
    val set = this.toMutableSet()
    set.removeAll(other)
}

```

Returns a set containing all elements that are contained by this array and not contained by the specified collection. The returned set preserves the element iteration order of the original array.

```

public infix fun LongArray.subtract(other: Iterable<Long>): Set<Long> {
    val set = this.toMutableSet()
    set.removeAll(other)
}

```

Returns a set containing all elements that are contained by this array and not contained by the specified collection. The returned set preserves the element iteration order of the original array.

```

public infix fun FloatArray.subtract(other: Iterable<Float>): Set<Float> {
    val set = this.toMutableSet()
    set.removeAll(other)
}

```

Returns a set containing all elements that are contained by this array and not contained by the specified collection. The returned set preserves the element iteration order of

the original array.

```

public infix fun DoubleArray.subtract(other: Iterable<Double>): Set<Double> {
    val set = this.toMutableSet()
    set.removeAll(other)
    return set
}

```

\* Returns a set containing all elements that are contained by this array and not contained by the specified collection.

\* The returned set preserves the element iteration order of the original array.

```

public infix fun BooleanArray.subtract(other: Iterable<Boolean>): Set<Boolean> {
    val set = this.toMutableSet()
    set.removeAll(other)
    return set
}

```

\* Returns a set containing all elements that are contained by this array and not contained by the specified collection.

\* The returned set preserves the element iteration order of the original array.

```

public infix fun CharArray.subtract(other: Iterable<Char>): Set<Char> {
    val set = this.toMutableSet()
    set.removeAll(other)
    return set
}

```

\* Returns a new [MutableSet] containing all distinct elements from the given array.

\* The returned set preserves the element iteration order of the original array.

```

public fun <T> Array<out T>.toMutableSet(): MutableSet<T> {
    return toCollection(LinkedHashSet<T>(mapCapacity(size)))
}

```

\* Returns a new [MutableSet] containing all distinct elements from the given array.

\* The returned set preserves the element iteration order of the original array.

```

public fun ByteArray.toMutableSet(): MutableSet<Byte> {
    return toCollection(LinkedHashSet<Byte>(mapCapacity(size)))
}

```

\* Returns a new [MutableSet] containing all distinct elements from the given array.

\* The returned set preserves the element iteration order of the original array.

```

public fun ShortArray.toMutableSet(): MutableSet<Short> {
    return toCollection(LinkedHashSet<Short>(mapCapacity(size)))
}

```

\* Returns a new [MutableSet] containing all distinct elements from the given array.

\* The returned set preserves the element iteration order of the original array.

```

public fun IntArray.toMutableSet(): MutableSet<Int> {
    return toCollection(LinkedHashSet<Int>(mapCapacity(size)))
}

```

\* Returns a new [MutableSet] containing all distinct elements from the given array.

\* The returned set preserves the element iteration order of the original array.

```

public fun LongArray.toMutableSet(): MutableSet<Long> {
    return toCollection(LinkedHashSet<Long>(mapCapacity(size)))
}

```

\* Returns a new [MutableSet] containing all distinct elements from the given array.

\* The returned set preserves the element iteration order of the original array.

```

public fun FloatArray.toMutableSet(): MutableSet<Float> {
    return toCollection(LinkedHashSet<Float>(mapCapacity(size)))
}

```

\* Returns a new [MutableSet] containing all distinct elements from the given array.

\* The returned set preserves the element iteration order of the original array.

```

public fun DoubleArray.toMutableSet(): MutableSet<Double> {
    return toCollection(LinkedHashSet<Double>(mapCapacity(size)))
}

```

\* Returns a new [MutableSet] containing all distinct elements from the given array.

\* The returned set preserves the element iteration order of the original array.

```

public fun BooleanArray.toMutableSet(): MutableSet<Boolean> {
    return toCollection(LinkedHashSet<Boolean>(mapCapacity(size)))
}

```

\* Returns a new [MutableSet] containing all distinct elements from the given array.

\* The returned set preserves the element iteration order of the original array.

```

public fun CharArray.toMutableSet(): MutableSet<Char> {
    return toCollection(LinkedHashSet<Char>(mapCapacity(size.coerceAtMost(128))))
}

```

\* Returns a set containing all distinct elements from both collections.

\* The returned set preserves the element iteration order of the original array.

\* Those elements of the [other] collection that are unique are iterated in the end in the order of the [other] collection.

\* To get a set containing all elements that are contained in both collections use [intersect].

```

public infix fun <T> Array<out T>.union(other: Iterable<T>): Set<T> {
    val set = this.toMutableSet()
    set.addAll(other)
    return set
}

```

\* Returns a set containing all distinct elements from both collections.

\* The returned set preserves the element iteration order of the original array.

\* Those elements of the [other] collection that are unique are iterated in the end in the order of the [other] collection.

\* To get a set containing all elements that are contained in both collections use [intersect].

```

public infix fun ByteArray.union(other: Iterable<Byte>): Set<Byte> {
    val set = this.toMutableSet()
    set.addAll(other)
    return set
}

```

\* Returns a set containing all distinct elements from both collections.

\* The returned set preserves the element iteration order of the original array.

\* Those elements of the [other] collection that are unique are iterated in the end in the order of the [other] collection.

\* To get a set containing all elements that are contained in both collections use [intersect].

```

public infix fun ShortArray.union(other:

```

```

Iterable<Short>): Set<Short> {\n  val set = this.toMutableSet()\n  set.addAll(other)\n  return set}\n\n/**\n * Returns a set containing all distinct elements from both collections.\n * \n * The returned set preserves the element iteration order of the original array.\n * Those elements of the [other] collection that are unique are iterated in the end\n * in the order of the [other] collection.\n * \n * To get a set containing all elements that are contained in both collections use [intersect].\n */\npublic infix fun IntArray.union(other: Iterable<Int>): Set<Int> {\n  val set = this.toMutableSet()\n  set.addAll(other)\n  return set}\n\n/**\n * Returns a set containing all distinct elements from both collections.\n * \n * The returned set preserves the element iteration order of the original array.\n * Those elements of the [other] collection that are unique are iterated in the end\n * in the order of the [other] collection.\n * \n * To get a set containing all elements that are contained in both collections use [intersect].\n */\npublic infix fun LongArray.union(other: Iterable<Long>): Set<Long> {\n  val set = this.toMutableSet()\n  set.addAll(other)\n  return set}\n\n/**\n * Returns a set containing all distinct elements from both collections.\n * \n * The returned set preserves the element iteration order of the original array.\n * Those elements of the [other] collection that are unique are iterated in the end\n * in the order of the [other] collection.\n * \n * To get a set containing all elements that are contained in both collections use [intersect].\n */\npublic infix fun FloatArray.union(other: Iterable<Float>): Set<Float> {\n  val set = this.toMutableSet()\n  set.addAll(other)\n  return set}\n\n/**\n * Returns a set containing all distinct elements from both collections.\n * \n * The returned set preserves the element iteration order of the original array.\n * Those elements of the [other] collection that are unique are iterated in the end\n * in the order of the [other] collection.\n * \n * To get a set containing all elements that are contained in both collections use [intersect].\n */\npublic infix fun DoubleArray.union(other: Iterable<Double>): Set<Double> {\n  val set = this.toMutableSet()\n  set.addAll(other)\n  return set}\n\n/**\n * Returns a set containing all distinct elements from both collections.\n * \n * The returned set preserves the element iteration order of the original array.\n * Those elements of the [other] collection that are unique are iterated in the end\n * in the order of the [other] collection.\n * \n * To get a set containing all elements that are contained in both collections use [intersect].\n */\npublic infix fun BooleanArray.union(other: Iterable<Boolean>): Set<Boolean> {\n  val set = this.toMutableSet()\n  set.addAll(other)\n  return set}\n\n/**\n * Returns a set containing all distinct elements from both collections.\n * \n * The returned set preserves the element iteration order of the original array.\n * Those elements of the [other] collection that are unique are iterated in the end\n * in the order of the [other] collection.\n * \n * To get a set containing all elements that are contained in both collections use [intersect].\n */\npublic infix fun CharArray.union(other: Iterable<Char>): Set<Char> {\n  val set = this.toMutableSet()\n  set.addAll(other)\n  return set}\n\n/**\n * Returns `true` if all elements match the given [predicate].\n * \n * @sample\n samples.collections.Collections.Aggregates.all\n */\npublic inline fun <T> Array<out T>.all(predicate: (T) -> Boolean): Boolean {\n  for (element in this) if (!predicate(element)) return false\n  return true}\n\n/**\n * Returns `true` if all elements match the given [predicate].\n * \n * @sample\n samples.collections.Collections.Aggregates.all\n */\npublic inline fun ByteArray.all(predicate: (Byte) -> Boolean): Boolean {\n  for (element in this) if (!predicate(element)) return false\n  return true}\n\n/**\n * Returns `true` if all elements match the given [predicate].\n * \n * @sample\n samples.collections.Collections.Aggregates.all\n */\npublic inline fun ShortArray.all(predicate: (Short) -> Boolean): Boolean {\n  for (element in this) if (!predicate(element)) return false\n  return true}\n\n/**\n * Returns `true` if all elements match the given [predicate].\n * \n * @sample\n samples.collections.Collections.Aggregates.all\n */\npublic inline fun IntArray.all(predicate: (Int) -> Boolean): Boolean {\n  for (element in this) if (!predicate(element)) return false\n  return true}\n\n/**\n * Returns `true` if all elements match the given [predicate].\n * \n * @sample\n samples.collections.Collections.Aggregates.all\n */\npublic inline fun LongArray.all(predicate: (Long) -> Boolean): Boolean {\n  for (element in this) if (!predicate(element)) return false\n  return true}\n\n/**\n * Returns `true` if all elements match the given [predicate].\n * \n * @sample\n samples.collections.Collections.Aggregates.all\n */\npublic inline fun FloatArray.all(predicate: (Float) -> Boolean): Boolean {\n  for (element in this) if (!predicate(element)) return false\n  return true}\n\n/**\n * Returns `true` if all elements match the given [predicate].\n * \n * @sample\n samples.collections.Collections.Aggregates.all\n */\npublic inline fun DoubleArray.all(predicate: (Double) -> Boolean): Boolean {\n  for (element in this) if (!predicate(element)) return

```

```

false\n    return true\n}\n\n/**\n * Returns `true` if all elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.all\n */\npublic inline fun BooleanArray.all(predicate: (Boolean) ->
Boolean): Boolean {\n    for (element in this) if (!predicate(element)) return false\n    return true\n}\n\n/**\n *
Returns `true` if all elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.all\n */\npublic inline fun CharArray.all(predicate: (Char) -> Boolean):
Boolean {\n    for (element in this) if (!predicate(element)) return false\n    return true\n}\n\n/**\n * Returns `true` if
array has at least one element.\n * \n * @sample samples.collections.Collections.Aggregates.any\n */\npublic fun
<T> Array<out T>.any(): Boolean {\n    return !isEmpty()\n}\n\n/**\n * Returns `true` if array has at least one
element.\n * \n * @sample samples.collections.Collections.Aggregates.any\n */\npublic fun ByteArray.any():
Boolean {\n    return !isEmpty()\n}\n\n/**\n * Returns `true` if array has at least one element.\n * \n * @sample
samples.collections.Collections.Aggregates.any\n */\npublic fun ShortArray.any(): Boolean {\n    return
!isEmpty()\n}\n\n/**\n * Returns `true` if array has at least one element.\n * \n * @sample
samples.collections.Collections.Aggregates.any\n */\npublic fun IntArray.any(): Boolean {\n    return
!isEmpty()\n}\n\n/**\n * Returns `true` if array has at least one element.\n * \n * @sample
samples.collections.Collections.Aggregates.any\n */\npublic fun LongArray.any(): Boolean {\n    return
!isEmpty()\n}\n\n/**\n * Returns `true` if array has at least one element.\n * \n * @sample
samples.collections.Collections.Aggregates.any\n */\npublic fun FloatArray.any(): Boolean {\n    return
!isEmpty()\n}\n\n/**\n * Returns `true` if array has at least one element.\n * \n * @sample
samples.collections.Collections.Aggregates.any\n */\npublic fun DoubleArray.any(): Boolean {\n    return
!isEmpty()\n}\n\n/**\n * Returns `true` if array has at least one element.\n * \n * @sample
samples.collections.Collections.Aggregates.any\n */\npublic fun BooleanArray.any(): Boolean {\n    return
!isEmpty()\n}\n\n/**\n * Returns `true` if array has at least one element.\n * \n * @sample
samples.collections.Collections.Aggregates.any\n */\npublic fun CharArray.any(): Boolean {\n    return
!isEmpty()\n}\n\n/**\n * Returns `true` if at least one element matches the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.anyWithPredicate\n */\npublic inline fun <T> Array<out
T>.any(predicate: (T) -> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return true\n    return
false\n}\n\n/**\n * Returns `true` if at least one element matches the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.anyWithPredicate\n */\npublic inline fun ByteArray.any(predicate:
(Byte) -> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return true\n    return
false\n}\n\n/**\n * Returns `true` if at least one element matches the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.anyWithPredicate\n */\npublic inline fun ShortArray.any(predicate:
(Short) -> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return true\n    return
false\n}\n\n/**\n * Returns `true` if at least one element matches the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.anyWithPredicate\n */\npublic inline fun IntArray.any(predicate: (Int) -
> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return true\n    return false\n}\n\n/**\n *
Returns `true` if at least one element matches the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.anyWithPredicate\n */\npublic inline fun LongArray.any(predicate:
(Long) -> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return true\n    return
false\n}\n\n/**\n * Returns `true` if at least one element matches the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.anyWithPredicate\n */\npublic inline fun FloatArray.any(predicate:
(Float) -> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return true\n    return
false\n}\n\n/**\n * Returns `true` if at least one element matches the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.anyWithPredicate\n */\npublic inline fun DoubleArray.any(predicate:
(Double) -> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return true\n    return
false\n}\n\n/**\n * Returns `true` if at least one element matches the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.anyWithPredicate\n */\npublic inline fun BooleanArray.any(predicate:
(Boolean) -> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return true\n    return
false\n}\n\n/**\n * Returns `true` if at least one element matches the given [predicate].\n * \n * @sample

```

```

samples.collections.Collections.Aggregates.anyWithPredicate\n *^public inline fun CharArray.any(predicate:
(Char) -> Boolean): Boolean {\n  for (element in this) if (predicate(element)) return true\n  return
false\n}\n\n/**\n * Returns the number of elements in this array.\n *^@kotlin.internal.InlineOnly\npublic inline
fun <T> Array<out T>.count(): Int {\n  return size\n}\n\n/**\n * Returns the number of elements in this array.\n
*^@kotlin.internal.InlineOnly\npublic inline fun ByteArray.count(): Int {\n  return size\n}\n\n/**\n * Returns the
number of elements in this array.\n *^@kotlin.internal.InlineOnly\npublic inline fun ShortArray.count(): Int {\n
return size\n}\n\n/**\n * Returns the number of elements in this array.\n *^@kotlin.internal.InlineOnly\npublic
inline fun IntArray.count(): Int {\n  return size\n}\n\n/**\n * Returns the number of elements in this array.\n
*^@kotlin.internal.InlineOnly\npublic inline fun LongArray.count(): Int {\n  return size\n}\n\n/**\n * Returns the
number of elements in this array.\n *^@kotlin.internal.InlineOnly\npublic inline fun FloatArray.count(): Int {\n
return size\n}\n\n/**\n * Returns the number of elements in this array.\n *^@kotlin.internal.InlineOnly\npublic
inline fun DoubleArray.count(): Int {\n  return size\n}\n\n/**\n * Returns the number of elements in this array.\n
*^@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.count(): Int {\n  return size\n}\n\n/**\n * Returns
the number of elements in this array.\n *^@kotlin.internal.InlineOnly\npublic inline fun CharArray.count(): Int {\n
return size\n}\n\n/**\n * Returns the number of elements matching the given [predicate].\n *^public inline fun
<T> Array<out T>.count(predicate: (T) -> Boolean): Int {\n  var count = 0\n  for (element in this) if
(predicate(element)) ++count\n  return count\n}\n\n/**\n * Returns the number of elements matching the given
[predicate].\n *^public inline fun ByteArray.count(predicate: (Byte) -> Boolean): Int {\n  var count = 0\n  for
(element in this) if (predicate(element)) ++count\n  return count\n}\n\n/**\n * Returns the number of elements
matching the given [predicate].\n *^public inline fun ShortArray.count(predicate: (Short) -> Boolean): Int {\n  var
count = 0\n  for (element in this) if (predicate(element)) ++count\n  return count\n}\n\n/**\n * Returns the
number of elements matching the given [predicate].\n *^public inline fun IntArray.count(predicate: (Int) ->
Boolean): Int {\n  var count = 0\n  for (element in this) if (predicate(element)) ++count\n  return
count\n}\n\n/**\n * Returns the number of elements matching the given [predicate].\n *^public inline fun
LongArray.count(predicate: (Long) -> Boolean): Int {\n  var count = 0\n  for (element in this) if
(predicate(element)) ++count\n  return count\n}\n\n/**\n * Returns the number of elements matching the given
[predicate].\n *^public inline fun FloatArray.count(predicate: (Float) -> Boolean): Int {\n  var count = 0\n  for
(element in this) if (predicate(element)) ++count\n  return count\n}\n\n/**\n * Returns the number of elements
matching the given [predicate].\n *^public inline fun DoubleArray.count(predicate: (Double) -> Boolean): Int {\n
var count = 0\n  for (element in this) if (predicate(element)) ++count\n  return count\n}\n\n/**\n * Returns the
number of elements matching the given [predicate].\n *^public inline fun BooleanArray.count(predicate: (Boolean)
-> Boolean): Int {\n  var count = 0\n  for (element in this) if (predicate(element)) ++count\n  return
count\n}\n\n/**\n * Returns the number of elements matching the given [predicate].\n *^public inline fun
CharArray.count(predicate: (Char) -> Boolean): Int {\n  var count = 0\n  for (element in this) if
(predicate(element)) ++count\n  return count\n}\n\n/**\n * Accumulates value starting with [initial] value and
applying [operation] from left to right\n * to current accumulator value and each element.\n * \n * Returns the
specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes current accumulator
value and an element, and calculates the next accumulator value.\n *^public inline fun <T, R> Array<out
T>.fold(initial: R, operation: (acc: R, T) -> R): R {\n  var accumulator = initial\n  for (element in this)
accumulator = operation(accumulator, element)\n  return accumulator\n}\n\n/**\n * Accumulates value starting
with [initial] value and applying [operation] from left to right\n * to current accumulator value and each element.\n
* \n * Returns the specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes
current accumulator value and an element, and calculates the next accumulator value.\n *^public inline fun <R>
ByteArray.fold(initial: R, operation: (acc: R, Byte) -> R): R {\n  var accumulator = initial\n  for (element in
this) accumulator = operation(accumulator, element)\n  return accumulator\n}\n\n/**\n * Accumulates value starting
with [initial] value and applying [operation] from left to right\n * to current accumulator value and each element.\n
* \n * Returns the specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes
current accumulator value and an element, and calculates the next accumulator value.\n *^public inline fun <R>

```

```

ShortArray.fold(initial: R, operation: (acc: R, Short) -> R): R {
    var accumulator = initial
    for (element in this)
        accumulator = operation(accumulator, element)
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from left to right to current accumulator value and each element. Returns the specified [initial] value if the array is empty. @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

```

public inline fun <R> IntArray.fold(initial: R, operation: (acc: R, Int) -> R): R {
    var accumulator = initial
    for (element in this)
        accumulator = operation(accumulator, element)
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from left to right to current accumulator value and each element. Returns the specified [initial] value if the array is empty. @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

```

public inline fun <R> LongArray.fold(initial: R, operation: (acc: R, Long) -> R): R {
    var accumulator = initial
    for (element in this)
        accumulator = operation(accumulator, element)
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from left to right to current accumulator value and each element. Returns the specified [initial] value if the array is empty. @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

```

public inline fun <R> FloatArray.fold(initial: R, operation: (acc: R, Float) -> R): R {
    var accumulator = initial
    for (element in this)
        accumulator = operation(accumulator, element)
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from left to right to current accumulator value and each element. Returns the specified [initial] value if the array is empty. @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

```

public inline fun <R> DoubleArray.fold(initial: R, operation: (acc: R, Double) -> R): R {
    var accumulator = initial
    for (element in this)
        accumulator = operation(accumulator, element)
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from left to right to current accumulator value and each element. Returns the specified [initial] value if the array is empty. @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

```

public inline fun <R> BooleanArray.fold(initial: R, operation: (acc: R, Boolean) -> R): R {
    var accumulator = initial
    for (element in this)
        accumulator = operation(accumulator, element)
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from left to right to current accumulator value and each element. Returns the specified [initial] value if the array is empty. @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

```

public inline fun <R> CharArray.fold(initial: R, operation: (acc: R, Char) -> R): R {
    var accumulator = initial
    for (element in this)
        accumulator = operation(accumulator, element)
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from left to right to current accumulator value and each element with its index in the original array. Returns the specified [initial] value if the array is empty. @param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

```

public inline fun <T, R> Array<out T>.foldIndexed(initial: R, operation: (index: Int, acc: R, T) -> R): R {
    var index = 0
    var accumulator = initial
    for (element in this)
        accumulator = operation(index++, accumulator, element)
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from left to right to current accumulator value and each element with its index in the original array. Returns the specified [initial] value if the array is empty. @param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

```

public inline fun <R> ByteArray.foldIndexed(initial: R, operation: (index: Int, acc: R, Byte) -> R): R {
    var index = 0
    var accumulator = initial
    for (element in this)
        accumulator = operation(index++, accumulator, element)
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from left to right to current accumulator value and each element with its index in the original array. Returns the specified [initial] value if the array is empty. @param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator

```

value.\n */\npublic inline fun <R> ShortArray.foldIndexed(initial: R, operation: (index: Int, acc: R, Short) -> R): R
{\n  var index = 0\n  var accumulator = initial\n  for (element in this) accumulator = operation(index++,
accumulator, element)\n  return accumulator}\n\n/**\n * Accumulates value starting with [initial] value and
applying [operation] from left to right\n * to current accumulator value and each element with its index in the
original array.\n * \n * Returns the specified [initial] value if the array is empty.\n * \n * @param [operation]
function that takes the index of an element, current accumulator value\n * and the element itself, and calculates the
next accumulator value.\n */\npublic inline fun <R> IntArray.foldIndexed(initial: R, operation: (index: Int, acc: R,
Int) -> R): R {\n  var index = 0\n  var accumulator = initial\n  for (element in this) accumulator =
operation(index++, accumulator, element)\n  return accumulator}\n\n/**\n * Accumulates value starting with [initial]
value and applying [operation] from left to right\n * to current accumulator value and each element with its
index in the original array.\n * \n * Returns the specified [initial] value if the array is empty.\n * \n * @param
[operation] function that takes the index of an element, current accumulator value\n * and the element itself, and
calculates the next accumulator value.\n */\npublic inline fun <R> LongArray.foldIndexed(initial: R, operation:
(index: Int, acc: R, Long) -> R): R {\n  var index = 0\n  var accumulator = initial\n  for (element in this)
accumulator = operation(index++, accumulator, element)\n  return accumulator}\n\n/**\n * Accumulates value
starting with [initial] value and applying [operation] from left to right\n * to current accumulator value and each
element with its index in the original array.\n * \n * Returns the specified [initial] value if the array is empty.\n
* \n * @param [operation] function that takes the index of an element, current accumulator value\n * and the element
itself, and calculates the next accumulator value.\n */\npublic inline fun <R> FloatArray.foldIndexed(initial: R,
operation: (index: Int, acc: R, Float) -> R): R {\n  var index = 0\n  var accumulator = initial\n  for (element in
this) accumulator = operation(index++, accumulator, element)\n  return accumulator}\n\n/**\n * Accumulates
value starting with [initial] value and applying [operation] from left to right\n * to current accumulator value and
each element with its index in the original array.\n * \n * Returns the specified [initial] value if the array is
empty.\n * \n * @param [operation] function that takes the index of an element, current accumulator value\n * and the
element itself, and calculates the next accumulator value.\n */\npublic inline fun <R>
DoubleArray.foldIndexed(initial: R, operation: (index: Int, acc: R, Double) -> R): R {\n  var index = 0\n  var
accumulator = initial\n  for (element in this) accumulator = operation(index++, accumulator, element)\n  return
accumulator}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from left to
right\n * to current accumulator value and each element with its index in the original array.\n * \n * Returns the
specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes the index of an
element, current accumulator value\n * and the element itself, and calculates the next accumulator value.\n */\npublic inline fun <R> BooleanArray.foldIndexed(initial: R, operation: (index: Int, acc: R, Boolean) -> R): R {\n
  var index = 0\n  var accumulator = initial\n  for (element in this) accumulator = operation(index++, accumulator,
element)\n  return accumulator}\n\n/**\n * Accumulates value starting with [initial] value and applying
[operation] from left to right\n * to current accumulator value and each element with its index in the original
array.\n * \n * Returns the specified [initial] value if the array is empty.\n * \n * @param [operation] function
that takes the index of an element, current accumulator value\n * and the element itself, and calculates the next
accumulator value.\n */\npublic inline fun <R> CharArray.foldIndexed(initial: R, operation: (index: Int, acc: R, Char) -> R): R
{\n  var index = 0\n  var accumulator = initial\n  for (element in this) accumulator = operation(index++,
accumulator, element)\n  return accumulator}\n\n/**\n * Accumulates value starting with [initial] value and
applying [operation] from right to left\n * to each element and current accumulator value.\n * \n * Returns the
specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes an element and
current accumulator value, and calculates the next accumulator value.\n */\npublic inline fun <T, R> Array<out
T>.foldRight(initial: R, operation: (T, acc: R) -> R): R {\n  var index = lastIndex\n  var accumulator = initial\n
while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n  return
accumulator}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from right to
left\n * to each element and current accumulator value.\n * \n * Returns the specified [initial] value if the array
is empty.\n * \n * @param [operation] function that takes an element and current accumulator value, and calculates
the

```

```

next accumulator value.\n */\npublic inline fun <R> ByteArray.foldRight(initial: R, operation: (Byte, acc: R) -> R):
R {\n  var index = lastIndex\n  var accumulator = initial\n  while (index >= 0) {\n    accumulator =
operation(get(index--), accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with
[initial] value and applying [operation] from right to left\n * to each element and current accumulator value.\n * \n *
Returns the specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes an element
and current accumulator value, and calculates the next accumulator value.\n */\npublic inline fun <R>
ShortArray.foldRight(initial: R, operation: (Short, acc: R) -> R): R {\n  var index = lastIndex\n  var accumulator =
initial\n  while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n  return
accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from right to
left\n * to each element and current accumulator value.\n * \n * Returns the specified [initial] value if the array is
empty.\n * \n * @param [operation] function that takes an element and current accumulator value, and calculates the
next accumulator value.\n */\npublic inline fun <R> IntArray.foldRight(initial: R, operation: (Int, acc: R) -> R): R
{\n  var index = lastIndex\n  var accumulator = initial\n  while (index >= 0) {\n    accumulator =
operation(get(index--), accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with
[initial] value and applying [operation] from right to left\n * to each element and current accumulator value.\n * \n *
Returns the specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes an element
and current accumulator value, and calculates the next accumulator value.\n */\npublic inline fun <R>
LongArray.foldRight(initial: R, operation: (Long, acc: R) -> R): R {\n  var index = lastIndex\n  var accumulator =
initial\n  while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n  return
accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from right to
left\n * to each element and current accumulator value.\n * \n * Returns the specified [initial] value if the array is
empty.\n * \n * @param [operation] function that takes an element and current accumulator value, and calculates the
next accumulator value.\n */\npublic inline fun <R> FloatArray.foldRight(initial: R, operation: (Float, acc: R) -> R):
R {\n  var index = lastIndex\n  var accumulator = initial\n  while (index >= 0) {\n    accumulator =
operation(get(index--), accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with
[initial] value and applying [operation] from right to left\n * to each element and current accumulator value.\n * \n *
Returns the specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes an element
and current accumulator value, and calculates the next accumulator value.\n */\npublic inline fun <R>
DoubleArray.foldRight(initial: R, operation: (Double, acc: R) -> R): R {\n  var index = lastIndex\n  var
accumulator = initial\n  while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n
return accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from right
to left\n * to each element and current accumulator value.\n * \n * Returns the specified [initial] value if the array is
empty.\n * \n * @param [operation] function that takes an element and current accumulator value, and calculates the
next accumulator value.\n */\npublic inline fun <R> BooleanArray.foldRight(initial: R, operation: (Boolean, acc: R)
-> R): R {\n  var index = lastIndex\n  var accumulator = initial\n  while (index >= 0) {\n    accumulator =
operation(get(index--), accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with
[initial] value and applying [operation] from right to left\n * to each element and current accumulator value.\n * \n *
Returns the specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes an element
and current accumulator value, and calculates the next accumulator value.\n */\npublic inline fun <R>
CharArray.foldRight(initial: R, operation: (Char, acc: R) -> R): R {\n  var index = lastIndex\n  var accumulator =
initial\n  while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n  return
accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from right to
left\n * to each element with its index in the original array and current accumulator value.\n * \n * Returns the
specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes the index of an
element, the element itself\n * and current accumulator value, and calculates the next accumulator value.\n */\npublic inline fun <T, R> Array<out T>.foldRightIndexed(initial: R, operation: (index: Int, T, acc: R) -> R): R
{\n  var index = lastIndex\n  var accumulator = initial\n  while (index >= 0) {\n    accumulator =
operation(index, get(index), accumulator)\n    --index\n  }\n  return accumulator\n}\n\n/**\n * Accumulates

```

value starting with [initial] value and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns the specified [initial] value if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value.

```

public inline fun <R> ByteArray.foldRightIndexed(initial: R,
operation: (index: Int, Byte, acc: R) -> R): R {
    var index = lastIndex
    var accumulator = initial
    while (index >= 0) {
        accumulator = operation(index, get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns the specified [initial] value if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value.

```

public inline fun <R> ShortArray.foldRightIndexed(initial: R, operation: (index: Int, Short, acc: R) -> R): R {
    var index = lastIndex
    var accumulator = initial
    while (index >= 0) {
        accumulator = operation(index,
get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns the specified [initial] value if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value.

```

public inline fun <R> IntArray.foldRightIndexed(initial: R, operation:
(index: Int, Int, acc: R) -> R): R {
    var index = lastIndex
    var accumulator = initial
    while (index >= 0) {
        accumulator = operation(index, get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns the specified [initial] value if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value.

```

public inline fun <R>
LongArray.foldRightIndexed(initial: R, operation: (index: Int, Long, acc: R) -> R): R {
    var index = lastIndex
    var accumulator = initial
    while (index >= 0) {
        accumulator = operation(index, get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns the specified [initial] value if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value.

```

public inline fun <R> FloatArray.foldRightIndexed(initial: R, operation: (index: Int, Float, acc: R) ->
R): R {
    var index = lastIndex
    var accumulator = initial
    while (index >= 0) {
        accumulator =
operation(index, get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns the specified [initial] value if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value.

```

public inline fun <R> DoubleArray.foldRightIndexed(initial:
R, operation: (index: Int, Double, acc: R) -> R): R {
    var index = lastIndex
    var accumulator = initial
    while (index >= 0) {
        accumulator = operation(index, get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns the specified [initial] value if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value.

```

public inline fun <R> BooleanArray.foldRightIndexed(initial: R, operation: (index: Int, Boolean, acc: R) -> R):
R {
    var index = lastIndex
    var accumulator = initial
    while (index >= 0) {
        accumulator =
operation(index, get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with [initial] value and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns the specified [initial] value if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator

```

value, and calculates the next accumulator value.\n */\npublic inline fun <R> CharArray.foldRightIndexed(initial: R,
operation: (index: Int, Char, acc: R) -> R): R {\n  var index = lastIndex\n  var accumulator = initial\n  while
(index >= 0) {\n    accumulator = operation(index, get(index), accumulator)\n    --index\n  }\n  return
accumulator\n}\n\n/**\n * Performs the given [action] on each element.\n */\npublic inline fun <T> Array<out
T>.forEach(action: (T) -> Unit): Unit {\n  for (element in this) action(element)\n}\n\n/**\n * Performs the given
[action] on each element.\n */\npublic inline fun ByteArray.forEach(action: (Byte) -> Unit): Unit {\n  for (element
in this) action(element)\n}\n\n/**\n * Performs the given [action] on each element.\n */\npublic inline fun
ShortArray.forEach(action: (Short) -> Unit): Unit {\n  for (element in this) action(element)\n}\n\n/**\n * Performs
the given [action] on each element.\n */\npublic inline fun IntArray.forEach(action: (Int) -> Unit): Unit {\n  for
(element in this) action(element)\n}\n\n/**\n * Performs the given [action] on each element.\n */\npublic inline fun
LongArray.forEach(action: (Long) -> Unit): Unit {\n  for (element in this) action(element)\n}\n\n/**\n * Performs
the given [action] on each element.\n */\npublic inline fun FloatArray.forEach(action: (Float) -> Unit): Unit {\n
for (element in this) action(element)\n}\n\n/**\n * Performs the given [action] on each element.\n */\npublic inline
fun DoubleArray.forEach(action: (Double) -> Unit): Unit {\n  for (element in this) action(element)\n}\n\n/**\n *
Performs the given [action] on each element.\n */\npublic inline fun BooleanArray.forEach(action: (Boolean) ->
Unit): Unit {\n  for (element in this) action(element)\n}\n\n/**\n * Performs the given [action] on each element.\n
*/\npublic inline fun CharArray.forEach(action: (Char) -> Unit): Unit {\n  for (element in this)
action(element)\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index with the
element.\n */\n * @param [action] function that takes the index of an element and the element itself\n * and performs the
action on the element.\n */\npublic inline fun <T> Array<out T>.forEachIndexed(action: (index: Int, T) -> Unit):
Unit {\n  var index = 0\n  for (item in this) action(index++, item)\n}\n\n/**\n * Performs the given [action] on
each element, providing sequential index with the element.\n */\n * @param [action] function that takes the index of an
element and the element itself\n * and performs the action on the element.\n */\npublic inline fun
ByteArray.forEachIndexed(action: (index: Int, Byte) -> Unit): Unit {\n  var index = 0\n  for (item in this)
action(index++, item)\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index with
the element.\n */\n * @param [action] function that takes the index of an element and the element itself\n * and performs
the action on the element.\n */\npublic inline fun ShortArray.forEachIndexed(action: (index: Int, Short) -> Unit):
Unit {\n  var index = 0\n  for (item in this) action(index++, item)\n}\n\n/**\n * Performs the given [action] on
each element, providing sequential index with the element.\n */\n * @param [action] function that takes the index of an
element and the element itself\n * and performs the action on the element.\n */\npublic inline fun
IntArray.forEachIndexed(action: (index: Int, Int) -> Unit): Unit {\n  var index = 0\n  for (item in this)
action(index++, item)\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index with
the element.\n */\n * @param [action] function that takes the index of an element and the element itself\n * and performs
the action on the element.\n */\npublic inline fun LongArray.forEachIndexed(action: (index: Int, Long) -> Unit):
Unit {\n  var index = 0\n  for (item in this) action(index++, item)\n}\n\n/**\n * Performs the given [action] on
each element, providing sequential index with the element.\n */\n * @param [action] function that takes the index of an
element and the element itself\n * and performs the action on the element.\n */\npublic inline fun
FloatArray.forEachIndexed(action: (index: Int, Float) -> Unit): Unit {\n  var index = 0\n  for (item in this)
action(index++, item)\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index with
the element.\n */\n * @param [action] function that takes the index of an element and the element itself\n * and performs
the action on the element.\n */\npublic inline fun DoubleArray.forEachIndexed(action: (index: Int, Double) -> Unit):
Unit {\n  var index = 0\n  for (item in this) action(index++, item)\n}\n\n/**\n * Performs the given [action] on
each element, providing sequential index with the element.\n */\n * @param [action] function that takes the index of an
element and the element itself\n * and performs the action on the element.\n */\npublic inline fun
BooleanArray.forEachIndexed(action: (index: Int, Boolean) -> Unit): Unit {\n  var index = 0\n  for (item in this)
action(index++, item)\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index with
the element.\n */\n * @param [action] function that takes the index of an element and the element itself\n * and performs
the action on the element.\n */\npublic inline fun CharArray.forEachIndexed(action: (index: Int, Char) -> Unit): Unit

```

```

{\n  var index = 0\n  for (item in this) action(index++, item)\n}\n\n@Deprecated("Use maxOrNull instead.",
ReplaceWith("this.maxOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\n@SinceKotlin("1.1")\npublic fun Array<out Double>.max(): Double? {\n  return
maxOrNull()\n}\n\n@Deprecated("Use maxOrNull instead.",
ReplaceWith("this.maxOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\n@SinceKotlin("1.1")\npublic fun Array<out Float>.max(): Float? {\n  return
maxOrNull()\n}\n\n@Deprecated("Use maxOrNull instead.",
ReplaceWith("this.maxOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\npublic fun <T : Comparable<T>> Array<out T>.max(): T? {\n  return
maxOrNull()\n}\n\n@Deprecated("Use maxOrNull instead.",
ReplaceWith("this.maxOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\npublic fun ByteArray.max(): Byte? {\n  return maxOrNull()\n}\n\n@Deprecated("Use
maxOrNull instead.", ReplaceWith("this.maxOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4",
errorSince = "1.5", hiddenSince = "1.6")\npublic fun ShortArray.max(): Short? {\n  return
maxOrNull()\n}\n\n@Deprecated("Use maxOrNull instead.",
ReplaceWith("this.maxOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\npublic fun IntArray.max(): Int? {\n  return maxOrNull()\n}\n\n@Deprecated("Use
maxOrNull instead.", ReplaceWith("this.maxOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4",
errorSince = "1.5", hiddenSince = "1.6")\npublic fun LongArray.max(): Long? {\n  return
maxOrNull()\n}\n\n@Deprecated("Use maxOrNull instead.",
ReplaceWith("this.maxOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\npublic fun FloatArray.max(): Float? {\n  return maxOrNull()\n}\n\n@Deprecated("Use
maxOrNull instead.", ReplaceWith("this.maxOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4",
errorSince = "1.5", hiddenSince = "1.6")\npublic fun DoubleArray.max(): Double? {\n  return
maxOrNull()\n}\n\n@Deprecated("Use maxOrNull instead.",
ReplaceWith("this.maxOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\npublic fun CharArray.max(): Char? {\n  return maxOrNull()\n}\n\n@Deprecated("Use
maxByOrNull instead.", ReplaceWith("this.maxByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince =
"1.4", errorSince = "1.5", hiddenSince = "1.6")\npublic inline fun <T, R : Comparable<R>> Array<out
T>.maxBy(selector: (T) -> R): T? {\n  return maxByOrNull(selector)\n}\n\n@Deprecated("Use maxByOrNull
instead.", ReplaceWith("this.maxByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4",
errorSince = "1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> ByteArray.maxBy(selector:
(Byte) -> R): Byte? {\n  return maxByOrNull(selector)\n}\n\n@Deprecated("Use maxByOrNull instead.",
ReplaceWith("this.maxByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> ShortArray.maxBy(selector: (Short) -> R):
Short? {\n  return maxByOrNull(selector)\n}\n\n@Deprecated("Use maxByOrNull instead.",
ReplaceWith("this.maxByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> IntArray.maxBy(selector: (Int) -> R): Int?
{\n  return maxByOrNull(selector)\n}\n\n@Deprecated("Use maxByOrNull instead.",
ReplaceWith("this.maxByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> LongArray.maxBy(selector: (Long) -> R):
Long? {\n  return maxByOrNull(selector)\n}\n\n@Deprecated("Use maxByOrNull instead.",
ReplaceWith("this.maxByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> FloatArray.maxBy(selector: (Float) -> R):
Float? {\n  return maxByOrNull(selector)\n}\n\n@Deprecated("Use maxByOrNull instead.",
ReplaceWith("this.maxByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> DoubleArray.maxBy(selector: (Double) ->
R): Double? {\n  return maxByOrNull(selector)\n}\n\n@Deprecated("Use maxByOrNull instead.",

```

```

ReplaceWith("\this.maxByOrNull(selector)")\n\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> BooleanArray.maxBy(selector: (Boolean) -
> R): Boolean? {\n    return maxByOrNull(selector)\n}\n\n@Deprecated("\Use maxByOrNull instead.\",
ReplaceWith("\this.maxByOrNull(selector)")\n\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> CharArray.maxBy(selector: (Char) -> R):
Char? {\n    return maxByOrNull(selector)\n}\n\n/**\n * Returns the first element yielding the largest value of the
given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.maxByOrNull\n * \n\n@SinceKotlin("1.4")\npublic inline fun <T, R :
Comparable<R>> Array<out T>.maxByOrNull(selector: (T) -> R): T? {\n    if (isEmpty()) return null\n    var
maxElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return maxElem\n    var maxVale =
selector(maxElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (maxVale < v)
{\n            maxElem = e\n            maxVale = v\n        }\n    }\n    return maxElem\n}\n\n/**\n * Returns the first
element yielding the largest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.maxByOrNull\n * \n\n@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> ByteArray.maxByOrNull(selector: (Byte) -> R): Byte? {\n    if (isEmpty()) return null\n    var
maxElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return maxElem\n    var maxVale =
selector(maxElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (maxVale < v)
{\n            maxElem = e\n            maxVale = v\n        }\n    }\n    return maxElem\n}\n\n/**\n * Returns the first
element yielding the largest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.maxByOrNull\n * \n\n@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> ShortArray.maxByOrNull(selector: (Short) -> R): Short? {\n    if (isEmpty()) return null\n    var
maxElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return maxElem\n    var maxVale =
selector(maxElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (maxVale < v)
{\n            maxElem = e\n            maxVale = v\n        }\n    }\n    return maxElem\n}\n\n/**\n * Returns the first
element yielding the largest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.maxByOrNull\n * \n\n@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> IntArray.maxByOrNull(selector: (Int) -> R): Int? {\n    if (isEmpty()) return null\n    var maxElem
= this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return maxElem\n    var maxVale =
selector(maxElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (maxVale < v)
{\n            maxElem = e\n            maxVale = v\n        }\n    }\n    return maxElem\n}\n\n/**\n * Returns the first
element yielding the largest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.maxByOrNull\n * \n\n@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> LongArray.maxByOrNull(selector: (Long) -> R): Long? {\n    if (isEmpty()) return null\n    var
maxElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return maxElem\n    var maxVale =
selector(maxElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (maxVale < v)
{\n            maxElem = e\n            maxVale = v\n        }\n    }\n    return maxElem\n}\n\n/**\n * Returns the first
element yielding the largest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.maxByOrNull\n * \n\n@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> FloatArray.maxByOrNull(selector: (Float) -> R): Float? {\n    if (isEmpty()) return null\n    var
maxElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return maxElem\n    var maxVale =
selector(maxElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (maxVale < v)
{\n            maxElem = e\n            maxVale = v\n        }\n    }\n    return maxElem\n}\n\n/**\n * Returns the first
element yielding the largest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.maxByOrNull\n * \n\n@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> DoubleArray.maxByOrNull(selector: (Double) -> R): Double? {\n    if (isEmpty()) return null\n    var
maxElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return maxElem\n    var maxVale =
selector(maxElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (maxVale < v)
{\n            maxElem = e\n            maxVale = v\n        }\n    }\n    return maxElem\n}\n\n/**\n * Returns the first

```

```

element yielding the largest value of the given function or `null` if there are no elements.
 * @sample samples.collections.Collections.Aggregates.maxByOrNull
 * @SinceKotlin("1.4")
public inline fun <R : Comparable<R>> BooleanArray.maxByOrNull(selector: (Boolean) -> R): Boolean? {
    if (isEmpty()) return null
    var maxElem = this[0]
    val lastIndex = this.lastIndex
    if (lastIndex == 0) return maxElem
    var maxValue = selector(maxElem)
    for (i in 1..lastIndex) {
        val e = this[i]
        val v = selector(e)
        if (maxValue < v) {
            maxElem = e
            maxValue = v
        }
    }
    return maxElem
}

Returns the first element yielding the largest value of the given function or `null` if there are no elements.
 * @sample samples.collections.Collections.Aggregates.maxByOrNull
 * @SinceKotlin("1.4")
public inline fun <R : Comparable<R>> CharArray.maxByOrNull(selector: (Char) -> R): Char? {
    if (isEmpty()) return null
    var maxElem = this[0]
    val lastIndex = this.lastIndex
    if (lastIndex == 0) return maxElem
    var maxValue = selector(maxElem)
    for (i in 1..lastIndex) {
        val e = this[i]
        val v = selector(e)
        if (maxValue < v) {
            maxElem = e
            maxValue = v
        }
    }
    return maxElem
}

Returns the largest value among all values produced by [selector] function applied to each element in the array.
 * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.
 * @throws NoSuchElementException if the array is empty.
 * @SinceKotlin("1.4")
 * @OptIn(kotlin.experimental.ExperimentalTypeInference::class)
 * @OverloadResolutionByLambdaReturnType
 * @kotlin.internal.InlineOnly
public inline fun <T> Array<out T>.maxOf(selector: (T) -> Double): Double {
    if (isEmpty()) throw NoSuchElementException()
    var maxValue = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        maxValue = maxOf(maxValue, v)
    }
    return maxValue
}

Returns the largest value among all values produced by [selector] function applied to each element in the array.
 * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.
 * @throws NoSuchElementException if the array is empty.
 * @SinceKotlin("1.4")
 * @OptIn(kotlin.experimental.ExperimentalTypeInference::class)
 * @OverloadResolutionByLambdaReturnType
 * @kotlin.internal.InlineOnly
public inline fun ByteArray.maxOf(selector: (Byte) -> Double): Double {
    if (isEmpty()) throw NoSuchElementException()
    var maxValue = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        maxValue = maxOf(maxValue, v)
    }
    return maxValue
}

Returns the largest value among all values produced by [selector] function applied to each element in the array.
 * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.
 * @throws NoSuchElementException if the array is empty.
 * @SinceKotlin("1.4")
 * @OptIn(kotlin.experimental.ExperimentalTypeInference::class)
 * @OverloadResolutionByLambdaReturnType
 * @kotlin.internal.InlineOnly
public inline fun ShortArray.maxOf(selector: (Short) -> Double): Double {
    if (isEmpty()) throw NoSuchElementException()
    var maxValue = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        maxValue = maxOf(maxValue, v)
    }
    return maxValue
}

Returns the largest value among all values produced by [selector] function applied to each element in the array.
 * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.
 * @throws NoSuchElementException if the array is empty.
 * @SinceKotlin("1.4")
 * @OptIn(kotlin.experimental.ExperimentalTypeInference::class)
 * @OverloadResolutionByLambdaReturnType
 * @kotlin.internal.InlineOnly
public inline fun IntArray.maxOf(selector: (Int) -> Double): Double {
    if (isEmpty()) throw NoSuchElementException()
    var maxValue = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        maxValue = maxOf(maxValue, v)
    }
    return maxValue
}

Returns the largest value among all values produced by [selector] function applied to each element in the array.
 * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.
 * @throws NoSuchElementException if the array is empty.
 * @SinceKotlin("1.4")
 * @OptIn(kotlin.experimental.ExperimentalTypeInference::class)
 * @OverloadResolutionByLambdaReturnType
 * @kotlin.internal.InlineOnly
public inline fun LongArray.maxOf(selector: (Long) -> Double): Double {
    if (isEmpty()) throw NoSuchElementException()
    var maxValue = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        maxValue = maxOf(maxValue, v)
    }
    return maxValue
}

Returns the largest value among all values produced by [selector] function applied to

```

each element in the array.\n \* \n \* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n \* \n \* @throws NoSuchElementException if the array is empty.\n

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.maxOf(selector: (Float) ->
Double): Double {\n if (isEmpty()) throw NoSuchElementException()\n var maxValue = selector(this[0])\n for
(i in 1..lastIndex) {\n val v = selector(this[i])\n maxValue = maxOf(maxValue, v)\n }\n return
maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.maxOf(selector: (Double) ->
Double): Double {\n if (isEmpty()) throw NoSuchElementException()\n var maxValue = selector(this[0])\n for
(i in 1..lastIndex) {\n val v = selector(this[i])\n maxValue = maxOf(maxValue, v)\n }\n return
maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.maxOf(selector: (Boolean) ->
Double): Double {\n if (isEmpty()) throw NoSuchElementException()\n var maxValue = selector(this[0])\n for
(i in 1..lastIndex) {\n val v = selector(this[i])\n maxValue = maxOf(maxValue, v)\n }\n return
maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.maxOf(selector: (Char) ->
Double): Double {\n if (isEmpty()) throw NoSuchElementException()\n var maxValue = selector(this[0])\n for
(i in 1..lastIndex) {\n val v = selector(this[i])\n maxValue = maxOf(maxValue, v)\n }\n return
maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out T>.maxOf(selector: (T) ->
Float): Float {\n if (isEmpty()) throw NoSuchElementException()\n var maxValue = selector(this[0])\n for (i
in 1..lastIndex) {\n val v = selector(this[i])\n maxValue = maxOf(maxValue, v)\n }\n return
maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.maxOf(selector: (Byte) -> Float):
Float {\n if (isEmpty()) throw NoSuchElementException()\n var maxValue = selector(this[0])\n for (i in
1..lastIndex) {\n val v = selector(this[i])\n maxValue = maxOf(maxValue, v)\n }\n return
maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.maxOf(selector: (Short) ->
Float): Float {\n if (isEmpty()) throw NoSuchElementException()\n var maxValue = selector(this[0])\n for (i
in 1..lastIndex) {\n val v = selector(this[i])\n maxValue = maxOf(maxValue, v)\n }\n return
maxValue\n}

```

```

maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.maxOf(selector: (Int) -> Float): Float {\n    if (isEmpty()) throw NoSuchElementException()\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.maxOf(selector: (Long) -> Float): Float {\n    if (isEmpty()) throw NoSuchElementException()\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.maxOf(selector: (Float) -> Float): Float {\n    if (isEmpty()) throw NoSuchElementException()\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.maxOf(selector: (Double) -> Float): Float {\n    if (isEmpty()) throw NoSuchElementException()\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.maxOf(selector: (Boolean) -> Float): Float {\n    if (isEmpty()) throw NoSuchElementException()\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.maxOf(selector: (Char) -> Float): Float {\n    if (isEmpty()) throw NoSuchElementException()\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Comparable<R>> Array<out T>.maxOf(selector: (T) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (maxValue < v) {\n
```

```

maxValue = v\n    }\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * @throws NoSuchElementException if the
array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
ByteArray.maxOf(selector: (Byte) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var
maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (maxValue < v) {\n
            maxValue = v\n        }\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values
produced by [selector] function\n * applied to each element in the array.\n * \n * @throws
NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
ShortArray.maxOf(selector: (Short) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var
maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (maxValue < v) {\n
            maxValue = v\n        }\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values
produced by [selector] function\n * applied to each element in the array.\n * \n * @throws
NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
IntArray.maxOf(selector: (Int) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var maxValue =
selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (maxValue < v) {\n
            maxValue = v\n        }\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * @throws NoSuchElementException if the
array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
LongArray.maxOf(selector: (Long) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var
maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (maxValue < v) {\n
            maxValue = v\n        }\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values
produced by [selector] function\n * applied to each element in the array.\n * \n * @throws
NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
FloatArray.maxOf(selector: (Float) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var
maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (maxValue < v) {\n
            maxValue = v\n        }\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values
produced by [selector] function\n * applied to each element in the array.\n * \n * @throws
NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
DoubleArray.maxOf(selector: (Double) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var
maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (maxValue < v) {\n
            maxValue = v\n        }\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values
produced by [selector] function\n * applied to each element in the array.\n * \n * @throws
NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
BooleanArray.maxOf(selector: (Boolean) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var

```

```

maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (maxValue < v) {\n      maxValue = v\n    }\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n */\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>> CharArray.maxOf(selector: (Char) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (maxValue < v) {\n      maxValue = v\n    }\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out T>.maxOfOrNull(selector: (T) -> Double): Double? {\n  if (isEmpty()) return null\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    maxValue = maxOf(maxValue, v)\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.maxOfOrNull(selector: (Byte) -> Double): Double? {\n  if (isEmpty()) return null\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    maxValue = maxOf(maxValue, v)\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.maxOfOrNull(selector: (Short) -> Double): Double? {\n  if (isEmpty()) return null\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    maxValue = maxOf(maxValue, v)\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.maxOfOrNull(selector: (Int) -> Double): Double? {\n  if (isEmpty()) return null\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    maxValue = maxOf(maxValue, v)\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.maxOfOrNull(selector: (Long) -> Double): Double? {\n  if (isEmpty()) return null\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    maxValue = maxOf(maxValue, v)\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.maxOfOrNull(selector: (Float) -

```

```
> Double): Double? {\n    if (isEmpty()) return null\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex)\n    {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\nReturns the largest value among all values produced by [selector] function\n* applied to each element in the array or `null` if there are no elements.\n* \n* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.maxOrNull(selector: (Double) -> Double): Double? {\n    if (isEmpty()) return null\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\nReturns the largest value among all values produced by [selector] function\n* applied to each element in the array or `null` if there are no elements.\n* \n* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.maxOrNull(selector: (Boolean) -> Double): Double? {\n    if (isEmpty()) return null\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\nReturns the largest value among all values produced by [selector] function\n* applied to each element in the array or `null` if there are no elements.\n* \n* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.maxOrNull(selector: (Char) -> Double): Double? {\n    if (isEmpty()) return null\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\nReturns the largest value among all values produced by [selector] function\n* applied to each element in the array or `null` if there are no elements.\n* \n* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out T>.maxOrNull(selector: (T) -> Float): Float? {\n    if (isEmpty()) return null\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\nReturns the largest value among all values produced by [selector] function\n* applied to each element in the array or `null` if there are no elements.\n* \n* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.maxOrNull(selector: (Byte) -> Float): Float? {\n    if (isEmpty()) return null\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\nReturns the largest value among all values produced by [selector] function\n* applied to each element in the array or `null` if there are no elements.\n* \n* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.maxOrNull(selector: (Short) -> Float): Float? {\n    if (isEmpty()) return null\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\nReturns the largest value among all values produced by [selector] function\n* applied to each element in the array or `null` if there are no elements.\n* \n* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
```

```

ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.maxOrNull(selector: (Int) ->
Float): Float? {\n  if (isEmpty()) return null\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n
val v = selector(this[i])\n    maxValue = maxOf(maxValue, v)\n  }\n  return maxValue\n}\n\n/**\n * Returns
the largest value among all values produced by [selector] function\n * applied to each element in the array or `null`
if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.maxOrNull(selector: (Long) -
> Float): Float? {\n  if (isEmpty()) return null\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n
val v = selector(this[i])\n    maxValue = maxOf(maxValue, v)\n  }\n  return maxValue\n}\n\n/**\n * Returns
the largest value among all values produced by [selector] function\n * applied to each element in the array or `null`
if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.maxOrNull(selector: (Float) -
> Float): Float? {\n  if (isEmpty()) return null\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n
val v = selector(this[i])\n    maxValue = maxOf(maxValue, v)\n  }\n  return maxValue\n}\n\n/**\n * Returns
the largest value among all values produced by [selector] function\n * applied to each element in the array or `null`
if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.maxOrNull(selector:
(Double) -> Float): Float? {\n  if (isEmpty()) return null\n  var maxValue = selector(this[0])\n  for (i in
1..lastIndex) {\n    val v = selector(this[i])\n    maxValue = maxOf(maxValue, v)\n  }\n  return
maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function
is `NaN`, the returned result is `NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.maxOrNull(selector:
(Boolean) -> Float): Float? {\n  if (isEmpty()) return null\n  var maxValue = selector(this[0])\n  for (i in
1..lastIndex) {\n    val v = selector(this[i])\n    maxValue = maxOf(maxValue, v)\n  }\n  return
maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function
is `NaN`, the returned result is `NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.maxOrNull(selector: (Char) ->
Float): Float? {\n  if (isEmpty()) return null\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n
val v = selector(this[i])\n    maxValue = maxOf(maxValue, v)\n  }\n  return maxValue\n}\n\n/**\n * Returns
the largest value among all values produced by [selector] function\n * applied to each element in the array or `null`
if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Comparable<R>> Array<out
T>.maxOrNull(selector: (T) -> R): R? {\n  if (isEmpty()) return null\n  var maxValue = selector(this[0])\n  for
(i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (maxValue < v) {\n      maxValue = v\n    }\n  }\n
return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n *
applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>

```

```

ByteArray.maxOrNull(selector: (Byte) -> R): R? {\n  if (isEmpty()) return null\n  var maxValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (maxValue < v) {\n
maxValue = v\n    }\n  }\n  return maxValue\n}\n\n * Returns the largest value among all values produced
by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
ShortArray.maxOrNull(selector: (Short) -> R): R? {\n  if (isEmpty()) return null\n  var maxValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (maxValue < v) {\n
maxValue = v\n    }\n  }\n  return maxValue\n}\n\n * Returns the largest value among all values produced
by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
IntArray.maxOrNull(selector: (Int) -> R): R? {\n  if (isEmpty()) return null\n  var maxValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (maxValue < v) {\n
maxValue = v\n    }\n  }\n  return maxValue\n}\n\n * Returns the largest value among all values produced
by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
LongArray.maxOrNull(selector: (Long) -> R): R? {\n  if (isEmpty()) return null\n  var maxValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (maxValue < v) {\n
maxValue = v\n    }\n  }\n  return maxValue\n}\n\n * Returns the largest value among all values produced
by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
FloatArray.maxOrNull(selector: (Float) -> R): R? {\n  if (isEmpty()) return null\n  var maxValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (maxValue < v) {\n
maxValue = v\n    }\n  }\n  return maxValue\n}\n\n * Returns the largest value among all values produced
by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
DoubleArray.maxOrNull(selector: (Double) -> R): R? {\n  if (isEmpty()) return null\n  var maxValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (maxValue < v) {\n
maxValue = v\n    }\n  }\n  return maxValue\n}\n\n * Returns the largest value among all values produced
by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
BooleanArray.maxOrNull(selector: (Boolean) -> R): R? {\n  if (isEmpty()) return null\n  var maxValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (maxValue < v) {\n
maxValue = v\n    }\n  }\n  return maxValue\n}\n\n * Returns the largest value among all values produced
by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
CharArray.maxOrNull(selector: (Char) -> R): R? {\n  if (isEmpty()) return null\n  var maxValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (maxValue < v) {\n
maxValue = v\n    }\n  }\n  return maxValue\n}\n\n * Returns the largest value according to the provided
[comparator]\n * among all values produced by [selector] function applied to each element in the array.\n * \n *
@throws NoSuchElementException if the array is empty.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution

```

```

ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R> Array<out
T>.maxOfWith(comparator: Comparator<in R>, selector: (T) -> R): R {\n  if (isEmpty()) throw
NoSuchElementException()\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v =
selector(this[i])\n    if (comparator.compare(maxValue, v) < 0) {\n      maxValue = v\n    }\n  }\n  return
maxValue\n}\n\n/**\n * Returns the largest value according to the provided [comparator]\n * among all values
produced by [selector] function applied to each element in the array.\n * \n * @throws NoSuchElementException if
the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> ByteArray.maxOfWith(comparator:
Comparator<in R>, selector: (Byte) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var
maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(maxValue, v) < 0) {\n      maxValue = v\n    }\n  }\n  return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> ShortArray.maxOfWith(comparator:
Comparator<in R>, selector: (Short) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var
maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(maxValue, v) < 0) {\n      maxValue = v\n    }\n  }\n  return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> IntArray.maxOfWith(comparator:
Comparator<in R>, selector: (Int) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var maxValue
= selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(maxValue, v) < 0) {\n      maxValue = v\n    }\n  }\n  return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> LongArray.maxOfWith(comparator:
Comparator<in R>, selector: (Long) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var
maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(maxValue, v) < 0) {\n      maxValue = v\n    }\n  }\n  return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> FloatArray.maxOfWith(comparator:
Comparator<in R>, selector: (Float) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var
maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(maxValue, v) < 0) {\n      maxValue = v\n    }\n  }\n  return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> DoubleArray.maxOfWith(comparator:
Comparator<in R>, selector: (Double) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var
maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(maxValue, v) < 0) {\n      maxValue = v\n    }\n  }\n  return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]

```

```

function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> BooleanArray.maxOfWith(comparator:
Comparator<in R>, selector: (Boolean) -> R): R {\n if (isEmpty()) throw NoSuchElementException()\n var
maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if
(comparator.compare(maxValue, v) < 0) {\n maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> CharArray.maxOfWith(comparator:
Comparator<in R>, selector: (Char) -> R): R {\n if (isEmpty()) throw NoSuchElementException()\n var
maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if
(comparator.compare(maxValue, v) < 0) {\n maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R> Array<out
T>.maxOfWithOrNull(comparator: Comparator<in R>, selector: (T) -> R): R? {\n if (isEmpty()) return null\n
var maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if
(comparator.compare(maxValue, v) < 0) {\n maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>
ByteArray.maxOfWithOrNull(comparator: Comparator<in R>, selector: (Byte) -> R): R? {\n if (isEmpty()) return
null\n var maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if
(comparator.compare(maxValue, v) < 0) {\n maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>
ShortArray.maxOfWithOrNull(comparator: Comparator<in R>, selector: (Short) -> R): R? {\n if (isEmpty())
return null\n var maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if
(comparator.compare(maxValue, v) < 0) {\n maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>
IntArray.maxOfWithOrNull(comparator: Comparator<in R>, selector: (Int) -> R): R? {\n if (isEmpty()) return
null\n var maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if
(comparator.compare(maxValue, v) < 0) {\n maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>
LongArray.maxOfWithOrNull(comparator: Comparator<in R>, selector: (Long) -> R): R? {\n if (isEmpty())
return null\n var maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if
(comparator.compare(maxValue, v) < 0) {\n maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n *
Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n

```

Returns the largest value according to the provided [comparator] \* among all values produced by [selector] function applied to each element in the array or `null` if there are no elements.

```

*^@SinceKotlin("1.4")^@OptIn(kotlin.experimental.ExperimentalTypeInference::class)^@OverloadResolution
ByLambdaReturnType^@kotlin.internal.InlineOnly^public inline fun <R>
FloatArray.maxOfWithOrNull(comparator: Comparator<in R>, selector: (Float) -> R): R? {
    if (isEmpty())
        return null
    var maxValue = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        if (comparator.compare(maxValue, v) < 0) {
            maxValue = v
        }
    }
    return maxValue
}

```

Returns the largest value according to the provided [comparator] \* among all values produced by [selector] function applied to each element in the array or `null` if there are no elements.

```

*^@SinceKotlin("1.4")^@OptIn(kotlin.experimental.ExperimentalTypeInference::class)^@OverloadResolution
ByLambdaReturnType^@kotlin.internal.InlineOnly^public inline fun <R>
DoubleArray.maxOfWithOrNull(comparator: Comparator<in R>, selector: (Double) -> R): R? {
    if (isEmpty())
        return null
    var maxValue = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        if (comparator.compare(maxValue, v) < 0) {
            maxValue = v
        }
    }
    return maxValue
}

```

Returns the largest value according to the provided [comparator] \* among all values produced by [selector] function applied to each element in the array or `null` if there are no elements.

```

*^@SinceKotlin("1.4")^@OptIn(kotlin.experimental.ExperimentalTypeInference::class)^@OverloadResolution
ByLambdaReturnType^@kotlin.internal.InlineOnly^public inline fun <R>
BooleanArray.maxOfWithOrNull(comparator: Comparator<in R>, selector: (Boolean) -> R): R? {
    if (isEmpty())
        return null
    var maxValue = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        if (comparator.compare(maxValue, v) < 0) {
            maxValue = v
        }
    }
    return maxValue
}

```

Returns the largest value according to the provided [comparator] \* among all values produced by [selector] function applied to each element in the array or `null` if there are no elements.

```

*^@SinceKotlin("1.4")^@OptIn(kotlin.experimental.ExperimentalTypeInference::class)^@OverloadResolution
ByLambdaReturnType^@kotlin.internal.InlineOnly^public inline fun <R>
CharArray.maxOfWithOrNull(comparator: Comparator<in R>, selector: (Char) -> R): R? {
    if (isEmpty())
        return null
    var maxValue = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        if (comparator.compare(maxValue, v) < 0) {
            maxValue = v
        }
    }
    return maxValue
}

```

Returns the largest element or `null` if there are no elements. \* If any of elements is `NaN` returns `NaN`.

```

*^@SinceKotlin("1.4")^public fun Array<out Double>.maxOrNull(): Double? {
    if (isEmpty())
        return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        max = maxOf(max, e)
    }
    return max
}

```

Returns the largest element or `null` if there are no elements. \* If any of elements is `NaN` returns `NaN`.

```

*^@SinceKotlin("1.4")^public fun Array<out Float>.maxOrNull(): Float? {
    if (isEmpty())
        return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        max = maxOf(max, e)
    }
    return max
}

```

Returns the largest element or `null` if there are no elements.

```

*^@SinceKotlin("1.4")^public fun <T : Comparable<T>> Array<out T>.maxOrNull(): T? {
    if (isEmpty())
        return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (max < e)
            max = e
    }
    return max
}

```

Returns the largest element or `null` if there are no elements.

```

*^@SinceKotlin("1.4")^public fun ByteArray.maxOrNull(): Byte? {
    if (isEmpty())
        return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (max < e)
            max = e
    }
    return max
}

```

Returns the largest element or `null` if there are no elements.

```

*^@SinceKotlin("1.4")^public fun ShortArray.maxOrNull(): Short? {
    if (isEmpty())
        return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (max < e)
            max = e
    }
    return max
}

```

Returns the largest element or `null` if there are no elements.

```

*^@SinceKotlin("1.4")^public fun IntArray.maxOrNull(): Int? {
    if (isEmpty())
        return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (max < e)
            max = e
    }
    return max
}

```

Returns the largest element or `null` if there are no elements.

```

*^@SinceKotlin("1.4")^public fun LongArray.maxOrNull(): Long? {
    if (isEmpty())
        return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (max < e)
            max = e
    }
    return max
}

```

```

* Returns the largest element or `null` if there are no elements.
 * If any of elements is `NaN` returns `NaN`.
@SinceKotlin("1.4")
public fun FloatArray.maxOrNull(): Float? {
    if (isEmpty()) return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        max = maxOf(max, e)
    }
    return max
}

* Returns the largest element or `null` if there are no elements.
 * If any of elements is `NaN` returns `NaN`.
@SinceKotlin("1.4")
public fun DoubleArray.maxOrNull(): Double? {
    if (isEmpty()) return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        max = maxOf(max, e)
    }
    return max
}

* Returns the largest element or `null` if there are no elements.
@SinceKotlin("1.4")
public fun CharArray.maxOrNull(): Char? {
    if (isEmpty()) return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (max < e) max = e
    }
    return max
}

@Deprecated("Use maxWithOrNull instead.",
ReplaceWith("this.maxWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun <T> Array<out T>.maxWith(comparator: Comparator<in T>): T? {
    return maxWithOrNull(comparator)
}

@Deprecated("Use maxWithOrNull instead.",
ReplaceWith("this.maxWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun ByteArray.maxWith(comparator: Comparator<in Byte>): Byte? {
    return maxWithOrNull(comparator)
}

@Deprecated("Use maxWithOrNull instead.",
ReplaceWith("this.maxWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun ShortArray.maxWith(comparator: Comparator<in Short>): Short? {
    return maxWithOrNull(comparator)
}

@Deprecated("Use maxWithOrNull instead.",
ReplaceWith("this.maxWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun IntArray.maxWith(comparator: Comparator<in Int>): Int? {
    return maxWithOrNull(comparator)
}

@Deprecated("Use maxWithOrNull instead.",
ReplaceWith("this.maxWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun LongArray.maxWith(comparator: Comparator<in Long>): Long? {
    return maxWithOrNull(comparator)
}

@Deprecated("Use maxWithOrNull instead.",
ReplaceWith("this.maxWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun FloatArray.maxWith(comparator: Comparator<in Float>): Float? {
    return maxWithOrNull(comparator)
}

@Deprecated("Use maxWithOrNull instead.",
ReplaceWith("this.maxWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun DoubleArray.maxWith(comparator: Comparator<in Double>): Double? {
    return maxWithOrNull(comparator)
}

@Deprecated("Use maxWithOrNull instead.",
ReplaceWith("this.maxWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun BooleanArray.maxWith(comparator: Comparator<in Boolean>): Boolean? {
    return maxWithOrNull(comparator)
}

@Deprecated("Use maxWithOrNull instead.",
ReplaceWith("this.maxWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun CharArray.maxWith(comparator: Comparator<in Char>): Char? {
    return maxWithOrNull(comparator)
}

* Returns the first element having the largest value according to the provided [comparator] or `null` if there are no elements.
@SinceKotlin("1.4")
public fun <T> Array<out T>.maxWithOrNull(comparator: Comparator<in T>): T? {
    if (isEmpty()) return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (comparator.compare(max, e) < 0) max = e
    }
    return max
}

* Returns the first element having the largest value according to the provided [comparator] or `null` if there are no elements.
@SinceKotlin("1.4")
public fun ByteArray.maxWithOrNull(comparator: Comparator<in Byte>): Byte? {
    if (isEmpty()) return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (comparator.compare(max, e) < 0) max = e
    }
    return max
}

* Returns the first element having the largest value according to the provided [comparator] or `null` if there are no elements.
@SinceKotlin("1.4")
public fun ShortArray.maxWithOrNull(comparator: Comparator<in Short>): Short? {
    if (isEmpty()) return null
    var max = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (comparator.compare(max, e) < 0) max = e
    }
    return max
}

* Returns the first element having the

```

```

largest value according to the provided [comparator] or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\npublic fun IntArray.maxWithOrNull(comparator: Comparator<in Int>): Int? {\n  if\n  (isEmpty()) return null\n  var max = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    if\n    (comparator.compare(max, e) < 0) max = e\n  }\n  return max\n}\n\n/**\n * Returns the first element having the\n largest value according to the provided [comparator] or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\npublic fun LongArray.maxWithOrNull(comparator: Comparator<in Long>): Long? {\n  if\n  (isEmpty()) return null\n  var max = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    if\n    (comparator.compare(max, e) < 0) max = e\n  }\n  return max\n}\n\n/**\n * Returns the first element having the\n largest value according to the provided [comparator] or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\npublic fun FloatArray.maxWithOrNull(comparator: Comparator<in Float>): Float? {\n  if\n  (isEmpty()) return null\n  var max = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    if\n    (comparator.compare(max, e) < 0) max = e\n  }\n  return max\n}\n\n/**\n * Returns the first element having the\n largest value according to the provided [comparator] or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\npublic fun DoubleArray.maxWithOrNull(comparator: Comparator<in Double>):\n Double? {\n  if (isEmpty()) return null\n  var max = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    if\n    (comparator.compare(max, e) < 0) max = e\n  }\n  return max\n}\n\n/**\n * Returns the first element having the\n largest value according to the provided [comparator] or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\npublic fun BooleanArray.maxWithOrNull(comparator: Comparator<in Boolean>):\n Boolean? {\n  if (isEmpty()) return null\n  var max = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    if\n    (comparator.compare(max, e) < 0) max = e\n  }\n  return max\n}\n\n/**\n * Returns the first element having the\n largest value according to the provided [comparator] or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\npublic fun CharArray.maxWithOrNull(comparator: Comparator<in Char>): Char? {\n  if\n  (isEmpty()) return null\n  var max = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    if\n    (comparator.compare(max, e) < 0) max = e\n  }\n  return max\n}\n\n@Deprecated("Use minOrNull instead.",\n ReplaceWith("this.minOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",\n hiddenSince = "1.6")\n@SinceKotlin("1.1")\npublic fun Array<out Double>.min(): Double? {\n  return\n  minOrNull()\n}\n\n@Deprecated("Use minOrNull instead.",\n ReplaceWith("this.minOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",\n hiddenSince = "1.6")\n@SinceKotlin("1.1")\npublic fun Array<out Float>.min(): Float? {\n  return\n  minOrNull()\n}\n\n@Deprecated("Use minOrNull instead.",\n ReplaceWith("this.minOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",\n hiddenSince = "1.6")\npublic fun <T : Comparable<T>> Array<out T>.min(): T? {\n  return\n  minOrNull()\n}\n\n@Deprecated("Use minOrNull instead.",\n ReplaceWith("this.minOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",\n hiddenSince = "1.6")\npublic fun ByteArray.min(): Byte? {\n  return minOrNull()\n}\n\n@Deprecated("Use\n minOrNull instead.", ReplaceWith("this.minOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4",\n errorSince = "1.5", hiddenSince = "1.6")\npublic fun ShortArray.min(): Short? {\n  return\n  minOrNull()\n}\n\n@Deprecated("Use minOrNull instead.",\n ReplaceWith("this.minOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",\n hiddenSince = "1.6")\npublic fun IntArray.min(): Int? {\n  return minOrNull()\n}\n\n@Deprecated("Use\n minOrNull instead.", ReplaceWith("this.minOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4",\n errorSince = "1.5", hiddenSince = "1.6")\npublic fun LongArray.min(): Long? {\n  return\n  minOrNull()\n}\n\n@Deprecated("Use minOrNull instead.",\n ReplaceWith("this.minOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",\n hiddenSince = "1.6")\npublic fun FloatArray.min(): Float? {\n  return minOrNull()\n}\n\n@Deprecated("Use\n minOrNull instead.", ReplaceWith("this.minOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4",\n errorSince = "1.5", hiddenSince = "1.6")\npublic fun DoubleArray.min(): Double? {\n  return\n  minOrNull()\n}\n\n@Deprecated("Use minOrNull instead.",

```

```

ReplaceWith("this.minOrNull()")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\npublic fun CharArray.min(): Char? {\n    return minOrNull()\n}\n\n@Deprecated("Use
minByOrNull instead.", ReplaceWith("this.minByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince =
"1.4", errorSince = "1.5", hiddenSince = "1.6")\npublic inline fun <T, R : Comparable<R>> Array<out
T>.minBy(selector: (T) -> R): T? {\n    return minByOrNull(selector)\n}\n\n@Deprecated("Use minByOrNull
instead.", ReplaceWith("this.minByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4",
errorSince = "1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> ByteArray.minBy(selector:
(Byte) -> R): Byte? {\n    return minByOrNull(selector)\n}\n\n@Deprecated("Use minByOrNull instead.",
ReplaceWith("this.minByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> ShortArray.minBy(selector: (Short) -> R):
Short? {\n    return minByOrNull(selector)\n}\n\n@Deprecated("Use minByOrNull instead.",
ReplaceWith("this.minByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> IntArray.minBy(selector: (Int) -> R): Int?
{\n    return minByOrNull(selector)\n}\n\n@Deprecated("Use minByOrNull instead.",
ReplaceWith("this.minByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> LongArray.minBy(selector: (Long) -> R):
Long? {\n    return minByOrNull(selector)\n}\n\n@Deprecated("Use minByOrNull instead.",
ReplaceWith("this.minByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> FloatArray.minBy(selector: (Float) -> R):
Float? {\n    return minByOrNull(selector)\n}\n\n@Deprecated("Use minByOrNull instead.",
ReplaceWith("this.minByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> DoubleArray.minBy(selector: (Double) ->
R): Double? {\n    return minByOrNull(selector)\n}\n\n@Deprecated("Use minByOrNull instead.",
ReplaceWith("this.minByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> BooleanArray.minBy(selector: (Boolean) -
> R): Boolean? {\n    return minByOrNull(selector)\n}\n\n@Deprecated("Use minByOrNull instead.",
ReplaceWith("this.minByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> CharArray.minBy(selector: (Char) -> R):
Char? {\n    return minByOrNull(selector)\n}\n\n/**\n * Returns the first element yielding the smallest value of the
given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.minByOrNull\n * \n *@SinceKotlin("1.4")\npublic inline fun <T, R :
Comparable<R>> Array<out T>.minByOrNull(selector: (T) -> R): T? {\n    if (isEmpty()) return null\n    var
minElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return minElem\n    var minValue =
selector(minElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (minValue > v)
{\n            minElem = e\n            minValue = v\n        }\n    }\n    return minElem\n}\n\n/**\n * Returns the first
element yielding the smallest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.minByOrNull\n * \n *@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> ByteArray.minByOrNull(selector: (Byte) -> R): Byte? {\n    if (isEmpty()) return null\n    var
minElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return minElem\n    var minValue =
selector(minElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (minValue > v)
{\n            minElem = e\n            minValue = v\n        }\n    }\n    return minElem\n}\n\n/**\n * Returns the first
element yielding the smallest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.minByOrNull\n * \n *@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> ShortArray.minByOrNull(selector: (Short) -> R): Short? {\n    if (isEmpty()) return null\n    var
minElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return minElem\n    var minValue =
selector(minElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (minValue > v)
{\n            minElem = e\n            minValue = v\n        }\n    }\n    return minElem\n}\n\n/**\n * Returns the first
element yielding the smallest value of the given function or `null` if there are no elements.\n * \n * @sample

```

```

samples.collections.Collections.Aggregates.minByOrNull\n *\n@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> IntArray.minByOrNull(selector: (Int) -> R): Int? {\n  if (isEmpty()) return null\n  var minElem
= this[0]\n  val lastIndex = this.lastIndex\n  if (lastIndex == 0) return minElem\n  var minValue =
selector(minElem)\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    val v = selector(e)\n    if (minValue > v)
{\n      minElem = e\n      minValue = v\n    }\n  }\n  return minElem\n}\n\n/**\n * Returns the first
element yielding the smallest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.minByOrNull\n *\n@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> LongArray.minByOrNull(selector: (Long) -> R): Long? {\n  if (isEmpty()) return null\n  var
minElem = this[0]\n  val lastIndex = this.lastIndex\n  if (lastIndex == 0) return minElem\n  var minValue =
selector(minElem)\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    val v = selector(e)\n    if (minValue > v)
{\n      minElem = e\n      minValue = v\n    }\n  }\n  return minElem\n}\n\n/**\n * Returns the first
element yielding the smallest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.minByOrNull\n *\n@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> FloatArray.minByOrNull(selector: (Float) -> R): Float? {\n  if (isEmpty()) return null\n  var
minElem = this[0]\n  val lastIndex = this.lastIndex\n  if (lastIndex == 0) return minElem\n  var minValue =
selector(minElem)\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    val v = selector(e)\n    if (minValue > v)
{\n      minElem = e\n      minValue = v\n    }\n  }\n  return minElem\n}\n\n/**\n * Returns the first
element yielding the smallest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.minByOrNull\n *\n@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> DoubleArray.minByOrNull(selector: (Double) -> R): Double? {\n  if (isEmpty()) return null\n
var minElem = this[0]\n  val lastIndex = this.lastIndex\n  if (lastIndex == 0) return minElem\n  var minValue =
selector(minElem)\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    val v = selector(e)\n    if (minValue > v)
{\n      minElem = e\n      minValue = v\n    }\n  }\n  return minElem\n}\n\n/**\n * Returns the first
element yielding the smallest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.minByOrNull\n *\n@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> BooleanArray.minByOrNull(selector: (Boolean) -> R): Boolean? {\n  if (isEmpty()) return
null\n  var minElem = this[0]\n  val lastIndex = this.lastIndex\n  if (lastIndex == 0) return minElem\n  var
minValue = selector(minElem)\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    val v = selector(e)\n    if
(minValue > v) {\n      minElem = e\n      minValue = v\n    }\n  }\n  return minElem\n}\n\n/**\n *
Returns the first element yielding the smallest value of the given function or `null` if there are no elements.\n *
\n * @sample samples.collections.Collections.Aggregates.minByOrNull\n *\n@SinceKotlin("1.4")\npublic inline fun
<R : Comparable<R>> CharArray.minByOrNull(selector: (Char) -> R): Char? {\n  if (isEmpty()) return null\n
var minElem = this[0]\n  val lastIndex = this.lastIndex\n  if (lastIndex == 0) return minElem\n  var minValue =
selector(minElem)\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    val v = selector(e)\n    if (minValue > v)
{\n      minElem = e\n      minValue = v\n    }\n  }\n  return minElem\n}\n\n/**\n * Returns the smallest
value among all values produced by [selector] function\n * applied to each element in the array.\n * \n * If any of
values produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws
NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out T>.minOf(selector: (T) ->
Double): Double {\n  if (isEmpty()) throw NoSuchElementException()\n  var minValue = selector(this[0])\n  for
(i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n  }\n  return
minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.minOf(selector: (Byte) ->
Double): Double {\n  if (isEmpty()) throw NoSuchElementException()\n  var minValue = selector(this[0])\n  for

```

```

(i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n } return
minValue}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.minOf(selector: (Short) ->
Double): Double {\n    if (isEmpty()) throw NoSuchElementException()\n    var minValue = selector(this[0])\n    for
(i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return
minValue}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.minOf(selector: (Int) -> Double):
Double {\n    if (isEmpty()) throw NoSuchElementException()\n    var minValue = selector(this[0])\n    for (i in
1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return
minValue}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.minOf(selector: (Long) ->
Double): Double {\n    if (isEmpty()) throw NoSuchElementException()\n    var minValue = selector(this[0])\n    for
(i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return
minValue}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.minOf(selector: (Float) ->
Double): Double {\n    if (isEmpty()) throw NoSuchElementException()\n    var minValue = selector(this[0])\n    for
(i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return
minValue}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.minOf(selector: (Double) ->
Double): Double {\n    if (isEmpty()) throw NoSuchElementException()\n    var minValue = selector(this[0])\n    for
(i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return
minValue}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.minOf(selector: (Boolean) ->
Double): Double {\n    if (isEmpty()) throw NoSuchElementException()\n    var minValue = selector(this[0])\n    for
(i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return
minValue}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.minOf(selector: (Char) ->

```

```

Double): Double {
    if (isEmpty()) throw NoSuchElementException()
    var minValue = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        minValue = minOf(minValue, v)
    }
    return
    minValue
}

/**
 * Returns the smallest value among all values produced by [selector] function
 * applied to each element in the array.
 * If any of values produced by [selector] function is `NaN`, the returned result is
 * `NaN`.
 * @throws NoSuchElementException if the array is empty.
 */
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@kotlin.internal.InlineOnly
public inline fun <T> Array<out T>.minOf(selector: (T) ->
Float): Float {
    if (isEmpty()) throw NoSuchElementException()
    var minValue = selector(this[0])
    for (i in
1..lastIndex) {
        val v = selector(this[i])
        minValue = minOf(minValue, v)
    }
    return
    minValue
}

/**
 * Returns the smallest value among all values produced by [selector] function
 * applied to each element in the array.
 * If any of values produced by [selector] function is `NaN`, the returned result is
 * `NaN`.
 * @throws NoSuchElementException if the array is empty.
 */
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@kotlin.internal.InlineOnly
public inline fun ByteArray.minOf(selector: (Byte) -> Float):
Float {
    if (isEmpty()) throw NoSuchElementException()
    var minValue = selector(this[0])
    for (i in
1..lastIndex) {
        val v = selector(this[i])
        minValue = minOf(minValue, v)
    }
    return
    minValue
}

/**
 * Returns the smallest value among all values produced by [selector] function
 * applied to each element in the array.
 * If any of values produced by [selector] function is `NaN`, the returned result is
 * `NaN`.
 * @throws NoSuchElementException if the array is empty.
 */
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@kotlin.internal.InlineOnly
public inline fun ShortArray.minOf(selector: (Short) ->
Float): Float {
    if (isEmpty()) throw NoSuchElementException()
    var minValue = selector(this[0])
    for (i in
1..lastIndex) {
        val v = selector(this[i])
        minValue = minOf(minValue, v)
    }
    return
    minValue
}

/**
 * Returns the smallest value among all values produced by [selector] function
 * applied to each element in the array.
 * If any of values produced by [selector] function is `NaN`, the returned result is
 * `NaN`.
 * @throws NoSuchElementException if the array is empty.
 */
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@kotlin.internal.InlineOnly
public inline fun IntArray.minOf(selector: (Int) -> Float):
Float {
    if (isEmpty()) throw NoSuchElementException()
    var minValue = selector(this[0])
    for (i in
1..lastIndex) {
        val v = selector(this[i])
        minValue = minOf(minValue, v)
    }
    return
    minValue
}

/**
 * Returns the smallest value among all values produced by [selector] function
 * applied to each element in the array.
 * If any of values produced by [selector] function is `NaN`, the returned result is
 * `NaN`.
 * @throws NoSuchElementException if the array is empty.
 */
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@kotlin.internal.InlineOnly
public inline fun LongArray.minOf(selector: (Long) ->
Float): Float {
    if (isEmpty()) throw NoSuchElementException()
    var minValue = selector(this[0])
    for (i in
1..lastIndex) {
        val v = selector(this[i])
        minValue = minOf(minValue, v)
    }
    return
    minValue
}

/**
 * Returns the smallest value among all values produced by [selector] function
 * applied to each element in the array.
 * If any of values produced by [selector] function is `NaN`, the returned result is
 * `NaN`.
 * @throws NoSuchElementException if the array is empty.
 */
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@kotlin.internal.InlineOnly
public inline fun FloatArray.minOf(selector: (Float) -> Float):
Float {
    if (isEmpty()) throw NoSuchElementException()
    var minValue = selector(this[0])
    for (i in
1..lastIndex) {
        val v = selector(this[i])
        minValue = minOf(minValue, v)
    }
    return
    minValue
}

/**
 * Returns the smallest value among all values produced by [selector] function
 * applied to each element in the array.
 * If any of values produced by [selector] function is `NaN`, the returned result is
 * `NaN`.
 * @throws NoSuchElementException if the array is empty.
 */
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution

```

```

ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.minOf(selector: (Double) ->
Float): Float {\n  if (isEmpty()) throw NoSuchElementException()\n  var minValue = selector(this[0])\n  for (i in
1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n  }\n  return
minValue}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.minOf(selector: (Boolean) ->
Float): Float {\n  if (isEmpty()) throw NoSuchElementException()\n  var minValue = selector(this[0])\n  for (i in
1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n  }\n  return
minValue}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.minOf(selector: (Char) -> Float):
Float {\n  if (isEmpty()) throw NoSuchElementException()\n  var minValue = selector(this[0])\n  for (i in
1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n  }\n  return
minValue}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Comparable<R>> Array<out
T>.minOf(selector: (T) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var minValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (minValue > v) {\n
minValue = v\n    }\n  }\n  return minValue}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * @throws NoSuchElementException if the
array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
ByteArray.minOf(selector: (Byte) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var minValue
= selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (minValue > v) {\n
minValue = v\n    }\n  }\n  return minValue}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * @throws NoSuchElementException if the
array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
ShortArray.minOf(selector: (Short) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var
minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (minValue > v) {\n
minValue = v\n    }\n  }\n  return minValue}\n\n/**\n * Returns the smallest value among all values
produced by [selector] function\n * applied to each element in the array.\n * \n * @throws
NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
IntArray.minOf(selector: (Int) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var minValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (minValue > v) {\n
minValue = v\n    }\n  }\n  return minValue}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * @throws NoSuchElementException if the
array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution

```

```

ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
LongArray.minOf(selector: (Long) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var
minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (minValue > v) {\n
    minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values
produced by [selector] function\n * applied to each element in the array.\n * \n * @throws
NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
FloatArray.minOf(selector: (Float) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var
minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (minValue > v) {\n
    minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values
produced by [selector] function\n * applied to each element in the array.\n * \n * @throws
NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
DoubleArray.minOf(selector: (Double) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var
minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (minValue > v) {\n
    minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values
produced by [selector] function\n * applied to each element in the array.\n * \n * @throws
NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
BooleanArray.minOf(selector: (Boolean) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var
minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (minValue > v) {\n
    minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values
produced by [selector] function\n * applied to each element in the array.\n * \n * @throws
NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
CharArray.minOf(selector: (Char) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var minValue
= selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (minValue > v) {\n
    minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of
values produced by [selector] function is `NaN`, the returned result is `NaN`.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out T>.minOfOrNull(selector:
(T) -> Double): Double? {\n  if (isEmpty()) return null\n  var minValue = selector(this[0])\n  for (i in
1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n  }\n  return
minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function
is `NaN`, the returned result is `NaN`.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.minOfOrNull(selector: (Byte) ->
Double): Double? {\n  if (isEmpty()) return null\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n
    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n  }\n  return minValue\n}\n\n/**\n * Returns
the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null`
if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
`NaN`.\n

```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.minOrNull(selector: (Short) -> Double): Double? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.minOrNull(selector: (Int) -> Double): Double? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.minOrNull(selector: (Long) -> Double): Double? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.minOrNull(selector: (Float) -> Double): Double? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.minOrNull(selector: (Double) -> Double): Double? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.minOrNull(selector: (Boolean) -> Double): Double? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.minOrNull(selector: (Char) -> Double): Double? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is
```

```
`NaN`.\n*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out T>.minOrNull(selector: (T) -> Float): Float? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * If any of values produced by [selector] function is `NaN`, the returned\n * result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.minOrNull(selector: (Byte) -> Float): Float? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * If any of values produced by [selector] function is `NaN`, the returned\n * result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.minOrNull(selector: (Short) -> Float): Float? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * If any of values produced by [selector] function is `NaN`, the returned\n * result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.minOrNull(selector: (Int) -> Float): Float? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * If any of values produced by [selector] function is `NaN`, the returned\n * result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.minOrNull(selector: (Long) -> Float): Float? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * If any of values produced by [selector] function is `NaN`, the returned\n * result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.minOrNull(selector: (Float) -> Float): Float? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * If any of values produced by [selector] function is `NaN`, the returned\n * result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.minOrNull(selector: (Double) -> Float): Float? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return\n    minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
```

each element in the array or `null` if there are no elements. \n \* \n \* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`. \n

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.minOfOrNull(selector:\n(Boolean) -> Float): Float? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in\n1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return\nminValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to\n * each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector] function\n * is `NaN`, the returned result is `NaN`. \n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.minOfOrNull(selector: (Char) ->\nFloat): Float? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the\n * smallest value among all values produced by [selector] function\n * applied to each element in the array or `null` if\n * there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Comparable<R>> Array<out\nT>.minOfOrNull(selector: (T) -> R): R? {\n    if (isEmpty()) return null\n    var minValue = selector(this[0])\n    for\n(i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (minValue > v) {\n            minValue = v\n        }\n    }\n    return\nminValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * \n * applied to each element in the array or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>\nByteArray.minOfOrNull(selector: (Byte) -> R): R? {\n    if (isEmpty()) return null\n    var minValue =\nselector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (minValue > v) {\n            minValue = v\n        }\n    }\n    return\nminValue\n}\n\n/**\n * Returns the smallest value among all values produced\n * by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>\nShortArray.minOfOrNull(selector: (Short) -> R): R? {\n    if (isEmpty()) return null\n    var minValue =\nselector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (minValue > v) {\n            minValue = v\n        }\n    }\n    return\nminValue\n}\n\n/**\n * Returns the smallest value among all values produced\n * by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>\nIntArray.minOfOrNull(selector: (Int) -> R): R? {\n    if (isEmpty()) return null\n    var minValue =\nselector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (minValue > v) {\n            minValue = v\n        }\n    }\n    return\nminValue\n}\n\n/**\n * Returns the smallest value among all values produced\n * by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>\nLongArray.minOfOrNull(selector: (Long) -> R): R? {\n    if (isEmpty()) return null\n    var minValue =\nselector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (minValue > v) {\n            minValue = v\n        }\n    }\n    return\nminValue\n}\n\n/**\n * Returns the smallest value among all values produced\n * by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>\nFloatArray.minOfOrNull(selector: (Float) -> R): R? {\n    if (isEmpty()) return null\n    var minValue =
```

```

selector(this[0])\n for (i in 1..lastIndex) {\n     val v = selector(this[i])\n     if (minValue > v) {\n
minValue = v\n     }\n }\n return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
DoubleArray.minOrNull(selector: (Double) -> R): R? {\n if (isEmpty()) return null\n var minValue =
selector(this[0])\n for (i in 1..lastIndex) {\n     val v = selector(this[i])\n     if (minValue > v) {\n
minValue = v\n     }\n }\n return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
BooleanArray.minOrNull(selector: (Boolean) -> R): R? {\n if (isEmpty()) return null\n var minValue =
selector(this[0])\n for (i in 1..lastIndex) {\n     val v = selector(this[i])\n     if (minValue > v) {\n
minValue = v\n     }\n }\n return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
CharArray.minOrNull(selector: (Char) -> R): R? {\n if (isEmpty()) return null\n var minValue =
selector(this[0])\n for (i in 1..lastIndex) {\n     val v = selector(this[i])\n     if (minValue > v) {\n
minValue = v\n     }\n }\n return minValue\n}\n\n/**\n * Returns the smallest value according to the provided
[comparator]\n * among all values produced by [selector] function applied to each element in the array.\n * \n *
@throws NoSuchElementException if the array is empty.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R> Array<out
T>.minOfWith(comparator: Comparator<in R>, selector: (T) -> R): R {\n if (isEmpty()) throw
NoSuchElementException()\n var minValue = selector(this[0])\n for (i in 1..lastIndex) {\n     val v =
selector(this[i])\n     if (comparator.compare(minValue, v) > 0) {\n         minValue = v\n     }\n }\n return
minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values
produced by [selector] function applied to each element in the array.\n * \n * @throws NoSuchElementException if
the array is empty.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> ByteArray.minOfWith(comparator:
Comparator<in R>, selector: (Byte) -> R): R {\n if (isEmpty()) throw NoSuchElementException()\n var
minValue = selector(this[0])\n for (i in 1..lastIndex) {\n     val v = selector(this[i])\n     if
(comparator.compare(minValue, v) > 0) {\n         minValue = v\n     }\n }\n return minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> ShortArray.minOfWith(comparator:
Comparator<in R>, selector: (Short) -> R): R {\n if (isEmpty()) throw NoSuchElementException()\n var
minValue = selector(this[0])\n for (i in 1..lastIndex) {\n     val v = selector(this[i])\n     if
(comparator.compare(minValue, v) > 0) {\n         minValue = v\n     }\n }\n return minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> IntArray.minOfWith(comparator:
Comparator<in R>, selector: (Int) -> R): R {\n if (isEmpty()) throw NoSuchElementException()\n var minValue
= selector(this[0])\n for (i in 1..lastIndex) {\n     val v = selector(this[i])\n     if (comparator.compare(minValue,

```

```

v) > 0) {\n        minValue = v\n    }\n }\n return minValue\n}\n\n/**\n * Returns the smallest value\n * according to the provided [comparator]\n * among all values produced by [selector] function applied to each\n * element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n *\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> LongArray.minOfWith(comparator:\nComparator<in R>, selector: (Long) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var\nminValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if\n(comparator.compare(minValue, v) > 0) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]\n * function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n *\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> FloatArray.minOfWith(comparator:\nComparator<in R>, selector: (Float) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var\nminValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if\n(comparator.compare(minValue, v) > 0) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]\n * function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n *\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> DoubleArray.minOfWith(comparator:\nComparator<in R>, selector: (Double) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var\nminValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if\n(comparator.compare(minValue, v) > 0) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]\n * function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n *\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> BooleanArray.minOfWith(comparator:\nComparator<in R>, selector: (Boolean) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var\nminValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if\n(comparator.compare(minValue, v) > 0) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]\n * function applied to each element in the array.\n * \n * @throws NoSuchElementException if the array is empty.\n *\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> CharArray.minOfWith(comparator:\nComparator<in R>, selector: (Char) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var\nminValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if\n(comparator.compare(minValue, v) > 0) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]\n * function applied to each element in the array or `null` if there are no elements.\n *\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R> Array<out\nT>.minOfWithOrNull(comparator: Comparator<in R>, selector: (T) -> R): R? {\n    if (isEmpty()) return null\n    var\nminValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if\n(comparator.compare(minValue, v) > 0) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]\n * function applied to each element in the array or `null` if there are no elements.\n *\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nByteArray.minOfWithOrNull(comparator: Comparator<in R>, selector: (Byte) -> R): R? {\n    if (isEmpty()) return

```

```

null\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(minValue, v) > 0) {\n      minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n *
Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>
ShortArray.minOfWithOrNull(comparator: Comparator<in R>, selector: (Short) -> R): R? {\n  if (isEmpty())
return null\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(minValue, v) > 0) {\n      minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n *
Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>
IntArray.minOfWithOrNull(comparator: Comparator<in R>, selector: (Int) -> R): R? {\n  if (isEmpty()) return
null\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(minValue, v) > 0) {\n      minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n *
Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>
LongArray.minOfWithOrNull(comparator: Comparator<in R>, selector: (Long) -> R): R? {\n  if (isEmpty()) return
null\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(minValue, v) > 0) {\n      minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n *
Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>
FloatArray.minOfWithOrNull(comparator: Comparator<in R>, selector: (Float) -> R): R? {\n  if (isEmpty()) return
null\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(minValue, v) > 0) {\n      minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n *
Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>
DoubleArray.minOfWithOrNull(comparator: Comparator<in R>, selector: (Double) -> R): R? {\n  if (isEmpty())
return null\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(minValue, v) > 0) {\n      minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n *
Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>
BooleanArray.minOfWithOrNull(comparator: Comparator<in R>, selector: (Boolean) -> R): R? {\n  if (isEmpty())
return null\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if
(comparator.compare(minValue, v) > 0) {\n      minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n *
Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>

```

```

CharArray.minOfWithOrNull(comparator: Comparator<in R>, selector: (Char) -> R): R? {
    if (isEmpty()) return null
    var minValue = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        if (comparator.compare(minValue, v) > 0) {
            minValue = v
        }
    }
    return minValue
}
Returns the smallest element or `null` if there are no elements.
* If any of elements is `NaN` returns `NaN`.
*/
@SinceKotlin("1.4")
public fun Array<out Double>.minOrNull(): Double? {
    if (isEmpty()) return null
    var min = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        min = minOf(min, e)
    }
    return min
}
Returns the smallest element or `null` if there are no elements.
* If any of elements is `NaN` returns `NaN`.
*/
@SinceKotlin("1.4")
public fun Array<out Float>.minOrNull(): Float? {
    if (isEmpty()) return null
    var min = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        min = minOf(min, e)
    }
    return min
}
Returns the smallest element or `null` if there are no elements.
*/
@SinceKotlin("1.4")
public fun <T : Comparable<T>> Array<out T>.minOrNull(): T? {
    if (isEmpty()) return null
    var min = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (min > e) min = e
    }
    return min
}
Returns the smallest element or `null` if there are no elements.
*/
@SinceKotlin("1.4")
public fun ByteArray.minOrNull(): Byte? {
    if (isEmpty()) return null
    var min = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (min > e) min = e
    }
    return min
}
Returns the smallest element or `null` if there are no elements.
*/
@SinceKotlin("1.4")
public fun ShortArray.minOrNull(): Short? {
    if (isEmpty()) return null
    var min = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (min > e) min = e
    }
    return min
}
Returns the smallest element or `null` if there are no elements.
*/
@SinceKotlin("1.4")
public fun IntArray.minOrNull(): Int? {
    if (isEmpty()) return null
    var min = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (min > e) min = e
    }
    return min
}
Returns the smallest element or `null` if there are no elements.
*/
@SinceKotlin("1.4")
public fun LongArray.minOrNull(): Long? {
    if (isEmpty()) return null
    var min = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (min > e) min = e
    }
    return min
}
Returns the smallest element or `null` if there are no elements.
* If any of elements is `NaN` returns `NaN`.
*/
@SinceKotlin("1.4")
public fun FloatArray.minOrNull(): Float? {
    if (isEmpty()) return null
    var min = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        min = minOf(min, e)
    }
    return min
}
Returns the smallest element or `null` if there are no elements.
* If any of elements is `NaN` returns `NaN`.
*/
@SinceKotlin("1.4")
public fun DoubleArray.minOrNull(): Double? {
    if (isEmpty()) return null
    var min = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        min = minOf(min, e)
    }
    return min
}
Returns the smallest element or `null` if there are no elements.
*/
@SinceKotlin("1.4")
public fun CharArray.minOrNull(): Char? {
    if (isEmpty()) return null
    var min = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (min > e) min = e
    }
    return min
}
@Deprecated("Use minWithOrNull instead.",
ReplaceWith("this.minWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun <T> Array<out T>.minWith(comparator: Comparator<in T>): T? {
    return minWithOrNull(comparator)
}
@Deprecated("Use minWithOrNull instead.",
ReplaceWith("this.minWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun ByteArray.minWith(comparator: Comparator<in Byte>): Byte? {
    return minWithOrNull(comparator)
}
@Deprecated("Use minWithOrNull instead.",
ReplaceWith("this.minWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun ShortArray.minWith(comparator: Comparator<in Short>): Short? {
    return minWithOrNull(comparator)
}
@Deprecated("Use minWithOrNull instead.",
ReplaceWith("this.minWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun IntArray.minWith(comparator: Comparator<in Int>): Int? {
    return minWithOrNull(comparator)
}
@Deprecated("Use minWithOrNull instead.",
ReplaceWith("this.minWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun LongArray.minWith(comparator: Comparator<in Long>): Long? {
    return minWithOrNull(comparator)
}
@Deprecated("Use minWithOrNull instead.",

```

```

ReplaceWith("\\this.minOrNull(comparator)")@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince
= \"1.5\", hiddenSince = \"1.6\")\npublic fun FloatArray.minWith(comparator: Comparator<in Float>): Float? {\n
return minOrNull(comparator)\n}\n\n@Deprecated(\"Use minOrNull instead.\",
ReplaceWith("\\this.minOrNull(comparator)")@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince
= \"1.5\", hiddenSince = \"1.6\")\npublic fun DoubleArray.minWith(comparator: Comparator<in Double>): Double?
{\n  return minOrNull(comparator)\n}\n\n@Deprecated(\"Use minOrNull instead.\",
ReplaceWith("\\this.minOrNull(comparator)")@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince
= \"1.5\", hiddenSince = \"1.6\")\npublic fun BooleanArray.minWith(comparator: Comparator<in Boolean>):
Boolean? {\n  return minOrNull(comparator)\n}\n\n@Deprecated(\"Use minOrNull instead.\",
ReplaceWith("\\this.minOrNull(comparator)")@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince
= \"1.5\", hiddenSince = \"1.6\")\npublic fun CharArray.minWith(comparator: Comparator<in Char>): Char? {\n
return minOrNull(comparator)\n}\n\n/**\n * Returns the first element having the smallest value according to
the provided [comparator] or `null` if there are no elements.\n * \n * @SinceKotlin(\"1.4\")\n * public fun <T> Array<out
T>.minWithOrNull(comparator: Comparator<in T>): T? {\n  if (isEmpty()) return null\n  var min = this[0]\n  for
(i in 1..lastIndex) {\n    val e = this[i]\n    if (comparator.compare(min, e) > 0) min = e\n  }\n  return
min\n}\n\n/**\n * Returns the first element having the smallest value according to the provided [comparator] or
`null` if there are no elements.\n * \n * @SinceKotlin(\"1.4\")\n * public fun ByteArray.minWithOrNull(comparator:
Comparator<in Byte>): Byte? {\n  if (isEmpty()) return null\n  var min = this[0]\n  for (i in 1..lastIndex) {\n
val e = this[i]\n    if (comparator.compare(min, e) > 0) min = e\n  }\n  return min\n}\n\n/**\n * Returns the first
element having the smallest value according to the provided [comparator] or `null` if there are no elements.\n
\n * \n * @SinceKotlin(\"1.4\")\n * public fun ShortArray.minWithOrNull(comparator: Comparator<in Short>): Short? {\n
if (isEmpty()) return null\n  var min = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    if
(comparator.compare(min, e) > 0) min = e\n  }\n  return min\n}\n\n/**\n * Returns the first element having the
smallest value according to the provided [comparator] or `null` if there are no elements.\n
\n * \n * @SinceKotlin(\"1.4\")\n * public fun IntArray.minWithOrNull(comparator: Comparator<in Int>): Int? {\n  if
(isEmpty()) return null\n  var min = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    if
(comparator.compare(min, e) > 0) min = e\n  }\n  return min\n}\n\n/**\n * Returns the first element having the
smallest value according to the provided [comparator] or `null` if there are no elements.\n
\n * \n * @SinceKotlin(\"1.4\")\n * public fun LongArray.minWithOrNull(comparator: Comparator<in Long>): Long? {\n
if (isEmpty()) return null\n  var min = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    if
(comparator.compare(min, e) > 0) min = e\n  }\n  return min\n}\n\n/**\n * Returns the first element having the
smallest value according to the provided [comparator] or `null` if there are no elements.\n
\n * \n * @SinceKotlin(\"1.4\")\n * public fun FloatArray.minWithOrNull(comparator: Comparator<in Float>): Float? {\n
if (isEmpty()) return null\n  var min = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    if
(comparator.compare(min, e) > 0) min = e\n  }\n  return min\n}\n\n/**\n * Returns the first element having the
smallest value according to the provided [comparator] or `null` if there are no elements.\n
\n * \n * @SinceKotlin(\"1.4\")\n * public fun DoubleArray.minWithOrNull(comparator: Comparator<in Double>):
Double? {\n  if (isEmpty()) return null\n  var min = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n
if
(comparator.compare(min, e) > 0) min = e\n  }\n  return min\n}\n\n/**\n * Returns the first element having the
smallest value according to the provided [comparator] or `null` if there are no elements.\n
\n * \n * @SinceKotlin(\"1.4\")\n * public fun BooleanArray.minWithOrNull(comparator: Comparator<in Boolean>):
Boolean? {\n  if (isEmpty()) return null\n  var min = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n
if
(comparator.compare(min, e) > 0) min = e\n  }\n  return min\n}\n\n/**\n * Returns the first element having the
smallest value according to the provided [comparator] or `null` if there are no elements.\n
\n * \n * @SinceKotlin(\"1.4\")\n * public fun CharArray.minWithOrNull(comparator: Comparator<in Char>): Char? {\n
if (isEmpty()) return null\n  var min = this[0]\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    if
(comparator.compare(min, e) > 0) min = e\n  }\n  return min\n}\n\n/**\n * Returns `true` if the array has no
elements.\n * \n * @sample samples.collections.Collections.Aggregates.none\n * \n * public fun <T> Array<out

```

```

T>.none(): Boolean {\n  return isEmpty()\n}\n\n/**\n * Returns `true` if the array has no elements.\n * \n *
@sample samples.collections.Collections.Aggregates.none\n */\npublic fun ByteArray.none(): Boolean {\n  return
isEmpty()\n}\n\n/**\n * Returns `true` if the array has no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.none\n */\npublic fun ShortArray.none(): Boolean {\n  return
isEmpty()\n}\n\n/**\n * Returns `true` if the array has no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.none\n */\npublic fun IntArray.none(): Boolean {\n  return
isEmpty()\n}\n\n/**\n * Returns `true` if the array has no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.none\n */\npublic fun LongArray.none(): Boolean {\n  return
isEmpty()\n}\n\n/**\n * Returns `true` if the array has no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.none\n */\npublic fun FloatArray.none(): Boolean {\n  return
isEmpty()\n}\n\n/**\n * Returns `true` if the array has no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.none\n */\npublic fun DoubleArray.none(): Boolean {\n  return
isEmpty()\n}\n\n/**\n * Returns `true` if the array has no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.none\n */\npublic fun BooleanArray.none(): Boolean {\n  return
isEmpty()\n}\n\n/**\n * Returns `true` if the array has no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.none\n */\npublic fun CharArray.none(): Boolean {\n  return
isEmpty()\n}\n\n/**\n * Returns `true` if no elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n */\npublic inline fun <T> Array<out
T>.none(predicate: (T) -> Boolean): Boolean {\n  for (element in this) if (predicate(element)) return false\n  return
true\n}\n\n/**\n * Returns `true` if no elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n */\npublic inline fun ByteArray.none(predicate:
(Byte) -> Boolean): Boolean {\n  for (element in this) if (predicate(element)) return false\n  return
true\n}\n\n/**\n * Returns `true` if no elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n */\npublic inline fun ShortArray.none(predicate:
(Short) -> Boolean): Boolean {\n  for (element in this) if (predicate(element)) return false\n  return
true\n}\n\n/**\n * Returns `true` if no elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n */\npublic inline fun IntArray.none(predicate: (Int)
-> Boolean): Boolean {\n  for (element in this) if (predicate(element)) return false\n  return true\n}\n\n/**\n *
Returns `true` if no elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n */\npublic inline fun LongArray.none(predicate:
(Long) -> Boolean): Boolean {\n  for (element in this) if (predicate(element)) return false\n  return
true\n}\n\n/**\n * Returns `true` if no elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n */\npublic inline fun FloatArray.none(predicate:
(Float) -> Boolean): Boolean {\n  for (element in this) if (predicate(element)) return false\n  return
true\n}\n\n/**\n * Returns `true` if no elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n */\npublic inline fun DoubleArray.none(predicate:
(Double) -> Boolean): Boolean {\n  for (element in this) if (predicate(element)) return false\n  return
true\n}\n\n/**\n * Returns `true` if no elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n */\npublic inline fun
BooleanArray.none(predicate: (Boolean) -> Boolean): Boolean {\n  for (element in this) if (predicate(element))
return false\n  return true\n}\n\n/**\n * Returns `true` if no elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n */\npublic inline fun CharArray.none(predicate:
(Char) -> Boolean): Boolean {\n  for (element in this) if (predicate(element)) return false\n  return
true\n}\n\n/**\n * Performs the given [action] on each element and returns the array itself afterwards.\n
*\n */\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out T>.onEach(action: (T) ->
Unit): Array<out T> {\n  return apply { for (element in this) action(element) }\n}\n\n/**\n * Performs the given
[action] on each element and returns the array itself afterwards.\n
*\n */\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.onEach(action: (Byte) ->

```

```

Unit): ByteArray {\n    return apply { for (element in this) action(element) }\n}\n\n/**\n * Performs the given
[action] on each element and returns the array itself afterwards.\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.onEach(action: (Short) ->
Unit): ShortArray {\n    return apply { for (element in this) action(element) }\n}\n\n/**\n * Performs the given
[action] on each element and returns the array itself afterwards.\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.onEach(action: (Int) -> Unit):
IntArray {\n    return apply { for (element in this) action(element) }\n}\n\n/**\n * Performs the given [action] on
each element and returns the array itself afterwards.\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.onEach(action: (Long) ->
Unit): LongArray {\n    return apply { for (element in this) action(element) }\n}\n\n/**\n * Performs the given
[action] on each element and returns the array itself afterwards.\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.onEach(action: (Float) ->
Unit): FloatArray {\n    return apply { for (element in this) action(element) }\n}\n\n/**\n * Performs the given
[action] on each element and returns the array itself afterwards.\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.onEach(action: (Double) ->
Unit): DoubleArray {\n    return apply { for (element in this) action(element) }\n}\n\n/**\n * Performs the given
[action] on each element and returns the array itself afterwards.\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.onEach(action: (Boolean)
-> Unit): BooleanArray {\n    return apply { for (element in this) action(element) }\n}\n\n/**\n * Performs the given
[action] on each element and returns the array itself afterwards.\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.onEach(action: (Char) ->
Unit): CharArray {\n    return apply { for (element in this) action(element) }\n}\n\n/**\n * Performs the given
[action] on each element, providing sequential index with the element,\n * and returns the array itself afterwards.\n *
@param [action] function that takes the index of an element and the element itself\n * and performs the action on
the element.\n *\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out
T>.onEachIndexed(action: (index: Int, T) -> Unit): Array<out T> {\n    return apply { forEachIndexed(action)
}\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index with the element,\n * and
returns the array itself afterwards.\n * @param [action] function that takes the index of an element and the element
itself\n * and performs the action on the element.\n *\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic
inline fun ByteArray.onEachIndexed(action: (index: Int, Byte) -> Unit): ByteArray {\n    return apply {
forEachIndexed(action) }\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index
with the element,\n * and returns the array itself afterwards.\n * @param [action] function that takes the index of an
element and the element itself\n * and performs the action on the element.\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.onEachIndexed(action:
(index: Int, Short) -> Unit): ShortArray {\n    return apply { forEachIndexed(action) }\n}\n\n/**\n * Performs the
given [action] on each element, providing sequential index with the element,\n * and returns the array itself
afterwards.\n * @param [action] function that takes the index of an element and the element itself\n * and performs
the action on the element.\n *\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun
IntArray.onEachIndexed(action: (index: Int, Int) -> Unit): IntArray {\n    return apply { forEachIndexed(action)
}\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index with the element,\n * and
returns the array itself afterwards.\n * @param [action] function that takes the index of an element and the element
itself\n * and performs the action on the element.\n *\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic
inline fun LongArray.onEachIndexed(action: (index: Int, Long) -> Unit): LongArray {\n    return apply {
forEachIndexed(action) }\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index
with the element,\n * and returns the array itself afterwards.\n * @param [action] function that takes the index of an
element and the element itself\n * and performs the action on the element.\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.onEachIndexed(action:
(index: Int, Float) -> Unit): FloatArray {\n    return apply { forEachIndexed(action) }\n}\n\n/**\n * Performs the

```

given [action] on each element, providing sequential index with the element, \n \* and returns the array itself afterwards. \n \* @param [action] function that takes the index of an element and the element itself \n \* and performs the action on the element. \n \*/\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.onEachIndexed(action: (index: Int, Double) -> Unit): DoubleArray {\n return apply { forEachIndexed(action) }\n}\n\n/\*\*\n \* Performs the given [action] on each element, providing sequential index with the element, \n \* and returns the array itself afterwards. \n \* @param [action] function that takes the index of an element and the element itself \n \* and performs the action on the element. \n \*/\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.onEachIndexed(action: (index: Int, Boolean) -> Unit): BooleanArray {\n return apply { forEachIndexed(action) }\n}\n\n/\*\*\n \* Performs the given [action] on each element, providing sequential index with the element, \n \* and returns the array itself afterwards. \n \* @param [action] function that takes the index of an element and the element itself \n \* and performs the action on the element. \n \*/\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.onEachIndexed(action: (index: Int, Char) -> Unit): CharArray {\n return apply { forEachIndexed(action) }\n}\n\n/\*\*\n \* Accumulates value starting with the first element and applying [operation] from left to right \n \* to current accumulator value and each element. \n \* \n \* Throws an exception if this array is empty. If the array can be empty in an expected way, \n \* please use [reduceOrNull] instead. It returns `null` when its receiver is empty. \n \* \n \* @param [operation] function that takes current accumulator value and an element, \n \* and calculates the next accumulator value. \n \* \n \* @sample samples.collections.Collections.Aggregates.reduce\n \*/\npublic inline fun <S, T : S> Array<out T>.reduce(operation: (acc: S, T) -> S): S {\n if (isEmpty())\n throw UnsupportedOperationException("Empty array can't be reduced.")\n var accumulator: S = this[0]\n for (index in 1..lastIndex) {\n accumulator = operation(accumulator, this[index])\n }\n return accumulator\n}\n\n/\*\*\n \* Accumulates value starting with the first element and applying [operation] from left to right \n \* to current accumulator value and each element. \n \* \n \* Throws an exception if this array is empty. If the array can be empty in an expected way, \n \* please use [reduceOrNull] instead. It returns `null` when its receiver is empty. \n \* \n \* @param [operation] function that takes current accumulator value and an element, \n \* and calculates the next accumulator value. \n \* \n \* @sample samples.collections.Collections.Aggregates.reduce\n \*/\npublic inline fun ByteArray.reduce(operation: (acc: Byte, Byte) -> Byte): Byte {\n if (isEmpty())\n throw UnsupportedOperationException("Empty array can't be reduced.")\n var accumulator = this[0]\n for (index in 1..lastIndex) {\n accumulator = operation(accumulator, this[index])\n }\n return accumulator\n}\n\n/\*\*\n \* Accumulates value starting with the first element and applying [operation] from left to right \n \* to current accumulator value and each element. \n \* \n \* Throws an exception if this array is empty. If the array can be empty in an expected way, \n \* please use [reduceOrNull] instead. It returns `null` when its receiver is empty. \n \* \n \* @param [operation] function that takes current accumulator value and an element, \n \* and calculates the next accumulator value. \n \* \n \* @sample samples.collections.Collections.Aggregates.reduce\n \*/\npublic inline fun ShortArray.reduce(operation: (acc: Short, Short) -> Short): Short {\n if (isEmpty())\n throw UnsupportedOperationException("Empty array can't be reduced.")\n var accumulator = this[0]\n for (index in 1..lastIndex) {\n accumulator = operation(accumulator, this[index])\n }\n return accumulator\n}\n\n/\*\*\n \* Accumulates value starting with the first element and applying [operation] from left to right \n \* to current accumulator value and each element. \n \* \n \* Throws an exception if this array is empty. If the array can be empty in an expected way, \n \* please use [reduceOrNull] instead. It returns `null` when its receiver is empty. \n \* \n \* @param [operation] function that takes current accumulator value and an element, \n \* and calculates the next accumulator value. \n \* \n \* @sample samples.collections.Collections.Aggregates.reduce\n \*/\npublic inline fun IntArray.reduce(operation: (acc: Int, Int) -> Int): Int {\n if (isEmpty())\n throw UnsupportedOperationException("Empty array can't be reduced.")\n var accumulator = this[0]\n for (index in 1..lastIndex) {\n accumulator = operation(accumulator, this[index])\n }\n return accumulator\n}\n\n/\*\*\n \* Accumulates value starting with the first element and applying [operation] from left to right \n \* to current accumulator value and each element. \n \* \n \* Throws an exception if this array is empty. If the array can be empty in an expected way, \n \* please use [reduceOrNull] instead. It returns `null` when its receiver is empty. \n \* \n \* @param [operation] function that takes current accumulator value and an element, \n \* and calculates the next accumulator value. \n \* \n \* @sample samples.collections.Collections.Aggregates.reduce\n \*/\npublic inline fun

@param [operation] function that takes current accumulator value and an element,  
 and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.reduce
public inline fun LongArray.reduce(operation: (acc: Long, Long) -> Long): Long {
    if (isEmpty()) throw UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator = operation(accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first element and applying [operation] from left to right to current accumulator value and each element. Throws an exception if this array is empty. If the array can be empty in an expected way, please use [reduceOrNull] instead. It returns `null` when its receiver is empty.

@param [operation] function that takes current accumulator value and an element,  
 and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.reduce
public inline fun FloatArray.reduce(operation: (acc: Float, Float) -> Float): Float {
    if (isEmpty()) throw UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator = operation(accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first element and applying [operation] from left to right to current accumulator value and each element. Throws an exception if this array is empty. If the array can be empty in an expected way, please use [reduceOrNull] instead. It returns `null` when its receiver is empty.

@param [operation] function that takes current accumulator value and an element,  
 and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.reduce
public inline fun DoubleArray.reduce(operation: (acc: Double, Double) -> Double): Double {
    if (isEmpty()) throw UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator = operation(accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first element and applying [operation] from left to right to current accumulator value and each element. Throws an exception if this array is empty. If the array can be empty in an expected way, please use [reduceOrNull] instead. It returns `null` when its receiver is empty.

@param [operation] function that takes current accumulator value and an element,  
 and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.reduce
public inline fun BooleanArray.reduce(operation: (acc: Boolean, Boolean) -> Boolean): Boolean {
    if (isEmpty()) throw UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator = operation(accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first element and applying [operation] from left to right to current accumulator value and each element. Throws an exception if this array is empty. If the array can be empty in an expected way, please use [reduceOrNull] instead. It returns `null` when its receiver is empty.

@param [operation] function that takes current accumulator value and an element,  
 and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.reduce
public inline fun CharArray.reduce(operation: (acc: Char, Char) -> Char): Char {
    if (isEmpty()) throw UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator = operation(accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first element and applying [operation] from left to right to current accumulator value and each element with its index in the original array. Throws an exception if this array is empty. If the array can be empty in an expected way, please use [reduceIndexedOrNull] instead. It returns `null` when its receiver is empty.

@param [operation] function that takes the index of an element, current accumulator value and the element itself,  
 and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.reduce
public inline fun <S, T : S> Array<out T>.reduceIndexed(operation: (index: Int, acc: S, T) -> S): S {
    if (isEmpty()) throw UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator: S = this[0]
    for (index in 1..lastIndex) {
        accumulator = operation(index, accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first element and applying [operation] from left to right to current accumulator value and each element with its index in the original array. Throws an

exception if this array is empty. If the array can be empty in an expected way, please use `[reduceIndexedOrNull]` instead. It returns `null` when its receiver is empty.

`@param [operation]` function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

`@sample samples.collections.Collections.Aggregates.reduce`

```
public inline fun
ByteArray.reduceIndexed(operation: (index: Int, acc: Byte, Byte) -> Byte): Byte {
    if (isEmpty()) throw
    UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in
    1..lastIndex) {
        accumulator = operation(index, accumulator, this[index])
    }
    return
    accumulator
}
```

Accumulates value starting with the first element and applying `[operation]` from left to right to current accumulator value and each element with its index in the original array. Throws an exception if this array is empty. If the array can be empty in an expected way, please use `[reduceIndexedOrNull]` instead. It returns `null` when its receiver is empty.

`@param [operation]` function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

`@sample samples.collections.Collections.Aggregates.reduce`

```
public inline fun
ShortArray.reduceIndexed(operation: (index: Int, acc: Short, Short) -> Short): Short {
    if (isEmpty()) throw
    UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in
    1..lastIndex) {
        accumulator = operation(index, accumulator, this[index])
    }
    return
    accumulator
}
```

Accumulates value starting with the first element and applying `[operation]` from left to right to current accumulator value and each element with its index in the original array. Throws an exception if this array is empty. If the array can be empty in an expected way, please use `[reduceIndexedOrNull]` instead. It returns `null` when its receiver is empty.

`@param [operation]` function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

`@sample samples.collections.Collections.Aggregates.reduce`

```
public inline fun
IntArray.reduceIndexed(operation: (index: Int, acc: Int, Int) -> Int): Int {
    if (isEmpty()) throw
    UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in
    1..lastIndex) {
        accumulator = operation(index, accumulator, this[index])
    }
    return
    accumulator
}
```

Accumulates value starting with the first element and applying `[operation]` from left to right to current accumulator value and each element with its index in the original array. Throws an exception if this array is empty. If the array can be empty in an expected way, please use `[reduceIndexedOrNull]` instead. It returns `null` when its receiver is empty.

`@param [operation]` function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

`@sample samples.collections.Collections.Aggregates.reduce`

```
public inline fun
LongArray.reduceIndexed(operation: (index: Int, acc: Long, Long) -> Long): Long {
    if (isEmpty()) throw
    UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in
    1..lastIndex) {
        accumulator = operation(index, accumulator, this[index])
    }
    return
    accumulator
}
```

Accumulates value starting with the first element and applying `[operation]` from left to right to current accumulator value and each element with its index in the original array. Throws an exception if this array is empty. If the array can be empty in an expected way, please use `[reduceIndexedOrNull]` instead. It returns `null` when its receiver is empty.

`@param [operation]` function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

`@sample samples.collections.Collections.Aggregates.reduce`

```
public inline fun
FloatArray.reduceIndexed(operation: (index: Int, acc: Float, Float) -> Float): Float {
    if (isEmpty()) throw
    UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in
    1..lastIndex) {
        accumulator = operation(index, accumulator, this[index])
    }
    return
    accumulator
}
```

Accumulates value starting with the first element and applying `[operation]` from left to right to current accumulator value and each element with its index in the original array. Throws an exception if this array is empty. If the array can be empty in an expected way, please use `[reduceIndexedOrNull]` instead. It returns `null` when its receiver is empty.

`@param [operation]` function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.reduce\n * \npublic inline fun
DoubleArray.reduceIndexed(operation: (index: Int, acc: Double, Double) -> Double): Double {\n if (isEmpty())\n
    throw UnsupportedOperationException("Empty array can't be reduced.")\n    var accumulator = this[0]\n    for
(index in 1..lastIndex) {\n        accumulator = operation(index, accumulator, this[index])\n    }\n    return
accumulator\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from left to
right\n * to current accumulator value and each element with its index in the original array.\n * \n * Throws an
exception if this array is empty. If the array can be empty in an expected way,\n * please use [reduceIndexedOrNull]
instead. It returns `null` when its receiver is empty.\n * \n * @param [operation] function that takes the index of an
element, current accumulator value and the element itself,\n * and calculates the next accumulator value.\n * \n *
@sample samples.collections.Collections.Aggregates.reduce\n * \npublic inline fun
BooleanArray.reduceIndexed(operation: (index: Int, acc: Boolean, Boolean) -> Boolean): Boolean {\n if
(isEmpty())\n    throw UnsupportedOperationException("Empty array can't be reduced.")\n    var accumulator =
this[0]\n    for (index in 1..lastIndex) {\n        accumulator = operation(index, accumulator, this[index])\n    }\n
return accumulator\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from
left to right\n * to current accumulator value and each element with its index in the original array.\n * \n * Throws
an exception if this array is empty. If the array can be empty in an expected way,\n * please use
[reduceIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n * @param [operation] function
that takes the index of an element, current accumulator value and the element itself,\n * and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduce\n * \npublic inline fun
CharArray.reduceIndexed(operation: (index: Int, acc: Char, Char) -> Char): Char {\n if (isEmpty())\n    throw
UnsupportedOperationException("Empty array can't be reduced.")\n    var accumulator = this[0]\n    for (index in
1..lastIndex) {\n        accumulator = operation(index, accumulator, this[index])\n    }\n    return
accumulator\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from left to
right\n * to current accumulator value and each element with its index in the original array.\n * \n * Returns `null` if
the array is empty.\n * \n * @param [operation] function that takes the index of an element, current accumulator
value and the element itself,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceOrNull\n * \n@SinceKotlin("1.4")\npublic inline fun <S, T : S>
Array<out T>.reduceIndexedOrNull(operation: (index: Int, acc: S, T) -> S): S? {\n if (isEmpty())\n    return
null\n    var accumulator: S = this[0]\n    for (index in 1..lastIndex) {\n        accumulator = operation(index,
accumulator, this[index])\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the first
element and applying [operation] from left to right\n * to current accumulator value and each element with its index
in the original array.\n * \n * Returns `null` if the array is empty.\n * \n * @param [operation] function that takes
the index of an element, current accumulator value and the element itself,\n * and calculates the next accumulator
value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*\n * \n@SinceKotlin("1.4")\npublic inline fun ByteArray.reduceIndexedOrNull(operation: (index: Int, acc: Byte,
Byte) -> Byte): Byte? {\n if (isEmpty())\n    return null\n    var accumulator = this[0]\n    for (index in
1..lastIndex) {\n        accumulator = operation(index, accumulator, this[index])\n    }\n    return
accumulator\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from left to
right\n * to current accumulator value and each element with its index in the original array.\n * \n * Returns `null`
if the array is empty.\n * \n * @param [operation] function that takes the index of an element, current accumulator
value and the element itself,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceOrNull\n * \n@SinceKotlin("1.4")\npublic inline fun
ShortArray.reduceIndexedOrNull(operation: (index: Int, acc: Short, Short) -> Short): Short? {\n if (isEmpty())\n
return null\n    var accumulator = this[0]\n    for (index in 1..lastIndex) {\n        accumulator = operation(index,
accumulator, this[index])\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the first
element and applying [operation] from left to right\n * to current accumulator value and each element with its index
in the original array.\n * \n * Returns `null` if the array is empty.\n * \n * @param [operation] function that takes
the index of an element, current accumulator value and the element itself,\n * and calculates the next accumulator

```

```

value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*\n@SinceKotlin("1.4")\npublic inline fun IntArray.reduceIndexedOrNull(operation: (index: Int, acc: Int, Int) ->
Int): Int? {\n if (isEmpty())\n return null\n var accumulator = this[0]\n for (index in 1..lastIndex) {\n
accumulator = operation(index, accumulator, this[index])\n }\n return accumulator\n}\n\n/**\n * Accumulates
value starting with the first element and applying [operation] from left to right\n * to current accumulator value and
each element with its index in the original array.\n * \n * Returns `null` if the array is empty.\n * \n * @param
[operation] function that takes the index of an element, current accumulator value and the element itself,\n * and
calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*\n@SinceKotlin("1.4")\npublic inline fun LongArray.reduceIndexedOrNull(operation: (index: Int, acc: Long,
Long) -> Long): Long? {\n if (isEmpty())\n return null\n var accumulator = this[0]\n for (index in
1..lastIndex) {\n accumulator = operation(index, accumulator, this[index])\n }\n return
accumulator\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from left to
right\n * to current accumulator value and each element with its index in the original array.\n * \n * Returns `null` if
the array is empty.\n * \n * @param [operation] function that takes the index of an element, current accumulator
value and the element itself,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceOrNull\n
*\n@SinceKotlin("1.4")\npublic inline fun FloatArray.reduceIndexedOrNull(operation: (index: Int, acc: Float, Float) -> Float): Float? {\n if (isEmpty())\n
return null\n var accumulator = this[0]\n for (index in 1..lastIndex) {\n accumulator = operation(index,
accumulator, this[index])\n }\n return accumulator\n}\n\n/**\n * Accumulates value starting with the first
element and applying [operation] from left to right\n * to current accumulator value and each element with its index
in the original array.\n * \n * Returns `null` if the array is empty.\n * \n * @param [operation] function that takes the
index of an element, current accumulator value and the element itself,\n * and calculates the next accumulator
value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*\n@SinceKotlin("1.4")\npublic inline fun DoubleArray.reduceIndexedOrNull(operation: (index: Int, acc:
Double, Double) -> Double): Double? {\n if (isEmpty())\n return null\n var accumulator = this[0]\n for
(index in 1..lastIndex) {\n accumulator = operation(index, accumulator, this[index])\n }\n return
accumulator\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from left to
right\n * to current accumulator value and each element with its index in the original array.\n * \n * Returns `null` if
the array is empty.\n * \n * @param [operation] function that takes the index of an element, current accumulator
value and the element itself,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceOrNull\n
*\n@SinceKotlin("1.4")\npublic inline fun BooleanArray.reduceIndexedOrNull(operation: (index: Int, acc: Boolean, Boolean) -> Boolean): Boolean? {\n if
(isEmpty())\n return null\n var accumulator = this[0]\n for (index in 1..lastIndex) {\n accumulator =
operation(index, accumulator, this[index])\n }\n return accumulator\n}\n\n/**\n * Accumulates value starting
with the first element and applying [operation] from left to right\n * to current accumulator value and each element
with its index in the original array.\n * \n * Returns `null` if the array is empty.\n * \n * @param [operation]
function that takes the index of an element, current accumulator value and the element itself,\n * and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*\n@SinceKotlin("1.4")\npublic inline fun CharArray.reduceIndexedOrNull(operation: (index: Int, acc: Char,
Char) -> Char): Char? {\n if (isEmpty())\n return null\n var accumulator = this[0]\n for (index in
1..lastIndex) {\n accumulator = operation(index, accumulator, this[index])\n }\n return
accumulator\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from left to
right\n * to current accumulator value and each element.\n * \n * Returns `null` if the array is empty.\n * \n *
@param [operation] function that takes current accumulator value and an element,\n * and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun <S, T : S>
Array<out T>.reduceOrNull(operation: (acc: S, T) -> S): S? {\n if (isEmpty())\n return null\n var
accumulator: S = this[0]\n for (index in 1..lastIndex) {\n accumulator = operation(accumulator, this[index])\n

```

```

}\n  return accumulator}\n\n/**\n * Accumulates value starting with the first element and applying [operation]
from left to right\n * to current accumulator value and each element.\n * \n * Returns `null` if the array is empty.\n *
\n * @param [operation] function that takes current accumulator value and an element,\n * and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*/\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
ByteArray.reduceOrNull(operation: (acc: Byte, Byte) -> Byte): Byte? {\n  if (isEmpty())\n    return null\n  var
accumulator = this[0]\n  for (index in 1..lastIndex) {\n    accumulator = operation(accumulator, this[index])\n
}\n  return accumulator}\n\n/**\n * Accumulates value starting with the first element and applying [operation]
from left to right\n * to current accumulator value and each element.\n * \n * Returns `null` if the array is empty.\n *
\n * @param [operation] function that takes current accumulator value and an element,\n * and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*/\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
ShortArray.reduceOrNull(operation: (acc: Short, Short) -> Short): Short? {\n  if (isEmpty())\n    return null\n
var accumulator = this[0]\n  for (index in 1..lastIndex) {\n    accumulator = operation(accumulator, this[index])\n
}\n  return accumulator}\n\n/**\n * Accumulates value starting with the first element and applying [operation]
from left to right\n * to current accumulator value and each element.\n * \n * Returns `null` if the array is empty.\n *
\n * @param [operation] function that takes current accumulator value and an element,\n * and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*/\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
IntArray.reduceOrNull(operation: (acc: Int, Int) -> Int): Int? {\n  if (isEmpty())\n    return null\n  var
accumulator = this[0]\n  for (index in 1..lastIndex) {\n    accumulator = operation(accumulator, this[index])\n
}\n  return accumulator}\n\n/**\n * Accumulates value starting with the first element and applying [operation]
from left to right\n * to current accumulator value and each element.\n * \n * Returns `null` if the array is empty.\n *
\n * @param [operation] function that takes current accumulator value and an element,\n * and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*/\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
LongArray.reduceOrNull(operation: (acc: Long, Long) -> Long): Long? {\n  if (isEmpty())\n    return null\n
var accumulator = this[0]\n  for (index in 1..lastIndex) {\n    accumulator = operation(accumulator, this[index])\n
}\n  return accumulator}\n\n/**\n * Accumulates value starting with the first element and applying [operation]
from left to right\n * to current accumulator value and each element.\n * \n * Returns `null` if the array is empty.\n *
\n * @param [operation] function that takes current accumulator value and an element,\n * and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*/\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
FloatArray.reduceOrNull(operation: (acc: Float, Float) -> Float): Float? {\n  if (isEmpty())\n    return null\n
var accumulator = this[0]\n  for (index in 1..lastIndex) {\n    accumulator = operation(accumulator, this[index])\n
}\n  return accumulator}\n\n/**\n * Accumulates value starting with the first element and applying [operation]
from left to right\n * to current accumulator value and each element.\n * \n * Returns `null` if the array is empty.\n *
\n * @param [operation] function that takes current accumulator value and an element,\n * and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*/\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
DoubleArray.reduceOrNull(operation: (acc: Double, Double) -> Double): Double? {\n  if (isEmpty())\n    return
null\n  var accumulator = this[0]\n  for (index in 1..lastIndex) {\n    accumulator = operation(accumulator,
this[index])\n  }\n  return accumulator}\n\n/**\n * Accumulates value starting with the first element and
applying [operation] from left to right\n * to current accumulator value and each element.\n * \n * Returns `null` if
the array is empty.\n * \n * @param [operation] function that takes current accumulator value and an element,\n *
and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceOrNull\n
*/\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun

```

```

BooleanArray.reduceOrNull(operation: (acc: Boolean, Boolean) -> Boolean): Boolean? {
    if (isEmpty())
        return null
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator =
            operation(accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first element and applying [operation] from left to right to current accumulator value and each element.

Returns `null` if the array is empty.

@param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

@sample

```

samples.collections.Collections.Aggregates.reduceOrNull

```

```

* Since Kotlin("1.4")
* WasExperimental(ExperimentalStdlibApi::class)
public inline fun
CharArray.reduceOrNull(operation: (acc: Char, Char) -> Char): Char? {
    if (isEmpty())
        return null
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator = operation(accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the last element and applying [operation] from right to left to each element and current accumulator value.

Throws an exception if this array is empty. If the array can be empty in an expected way, please use [reduceRightOrNull] instead. It returns `null` when its receiver is empty.

@param [operation] function that takes an element and current accumulator value, and calculates the next accumulator value.

@sample

```

samples.collections.Collections.Aggregates.reduceRight

```

```

public inline fun <S, T : S> Array<out T>.reduceRight(operation: (T, acc: S) -> S): S {
    var index = lastIndex
    if (index < 0) throw
        UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator: S = get(index--)
    while (index >= 0) {
        accumulator = operation(get(index--), accumulator)
    }
    return accumulator
}

```

Accumulates value starting with the last element and applying [operation] from right to left to each element and current accumulator value.

Throws an exception if this array is empty. If the array can be empty in an expected way, please use [reduceRightOrNull] instead. It returns `null` when its receiver is empty.

@param [operation] function that takes an element and current accumulator value, and calculates the next accumulator value.

@sample

```

samples.collections.Collections.Aggregates.reduceRight

```

```

public inline fun ByteArray.reduceRight(operation: (Byte, acc: Byte) -> Byte): Byte {
    var index = lastIndex
    if (index < 0)
        throw UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = get(index--)
    while (index >= 0) {
        accumulator = operation(get(index--), accumulator)
    }
    return accumulator
}

```

Accumulates value starting with the last element and applying [operation] from right to left to each element and current accumulator value.

Throws an exception if this array is empty. If the array can be empty in an expected way, please use [reduceRightOrNull] instead. It returns `null` when its receiver is empty.

@param [operation] function that takes an element and current accumulator value, and calculates the next accumulator value.

@sample

```

samples.collections.Collections.Aggregates.reduceRight

```

```

public inline fun ShortArray.reduceRight(operation: (Short, acc: Short) -> Short): Short {
    var index = lastIndex
    if (index < 0) throw
        UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = get(index--)
    while (index >= 0) {
        accumulator = operation(get(index--), accumulator)
    }
    return accumulator
}

```

Accumulates value starting with the last element and applying [operation] from right to left to each element and current accumulator value.

Throws an exception if this array is empty. If the array can be empty in an expected way, please use [reduceRightOrNull] instead. It returns `null` when its receiver is empty.

@param [operation] function that takes an element and current accumulator value, and calculates the next accumulator value.

@sample

```

samples.collections.Collections.Aggregates.reduceRight

```

```

public inline fun IntArray.reduceRight(operation: (Int, acc: Int) -> Int): Int {
    var index = lastIndex
    if (index < 0) throw
        UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = get(index--)
    while (index >= 0) {
        accumulator = operation(get(index--), accumulator)
    }
    return accumulator
}

```

Accumulates value starting with the last element and applying [operation] from right to left to each element and current accumulator value.

Throws an exception if this array is empty. If the array can be empty in an expected way, please use [reduceRightOrNull] instead. It returns `null` when its receiver is empty.

@param [operation] function that takes an element and current accumulator value, and calculates the next

```

accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceRight\n */\npublic inline
fun LongArray.reduceRight(operation: (Long, acc: Long) -> Long): Long {\n    var index = lastIndex\n    if (index <
0) throw UnsupportedOperationException("Empty array can't be reduced.")\n    var accumulator = get(index--)\n
while (index >= 0) {\n        accumulator = operation(get(index--), accumulator)\n    }\n    return
accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to
left\n * to each element and current accumulator value.\n * \n * Throws an exception if this array is empty. If
the array can be empty in an expected way,\n * please use [reduceRightOrNull] instead. It returns `null` when its
receiver is empty.\n * \n * @param [operation] function that takes an element and current accumulator value,\n *
and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n */\npublic inline fun FloatArray.reduceRight(operation:
(Float, acc: Float) -> Float): Float {\n    var index = lastIndex\n    if (index < 0) throw
UnsupportedOperationException("Empty array can't be reduced.")\n    var accumulator = get(index--)\n    while
(index >= 0) {\n        accumulator = operation(get(index--), accumulator)\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to left\n * to each element and
current accumulator value.\n * \n * Throws an exception if this array is empty. If the array can be empty in an
expected way,\n * please use [reduceRightOrNull] instead. It returns `null` when its receiver is empty.\n * \n *
@param [operation] function that takes an element and current accumulator value,\n * and calculates the next
accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n */\npublic inline fun DoubleArray.reduceRight(operation:
(Double, acc: Double) -> Double): Double {\n    var index = lastIndex\n    if (index < 0) throw
UnsupportedOperationException("Empty array can't be reduced.")\n    var accumulator =
get(index--)\n    while (index >= 0) {\n        accumulator = operation(get(index--), accumulator)\n    }\n    return
accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to
left\n * to each element and current accumulator value.\n * \n * Throws an exception if this array is empty. If
the array can be empty in an expected way,\n * please use [reduceRightOrNull] instead. It returns `null` when its
receiver is empty.\n * \n * @param [operation] function that takes an element and current accumulator value,\n *
and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n */\npublic inline fun
BooleanArray.reduceRight(operation: (Boolean, acc: Boolean) -> Boolean): Boolean {\n    var index = lastIndex\n
if (index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n    var accumulator =
get(index--)\n    while (index >= 0) {\n        accumulator = operation(get(index--), accumulator)\n    }\n    return
accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to
left\n * to each element and current accumulator value.\n * \n * Throws an exception if this array is empty. If
the array can be empty in an expected way,\n * please use [reduceRightOrNull] instead. It returns `null` when its
receiver is empty.\n * \n * @param [operation] function that takes an element and current accumulator value,\n *
and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n */\npublic inline fun
CharArray.reduceRight(operation:
(Char, acc: Char) -> Char): Char {\n    var index = lastIndex\n    if (index < 0) throw
UnsupportedOperationException("Empty array can't be reduced.")\n    var accumulator = get(index--)\n    while
(index >= 0) {\n        accumulator = operation(get(index--), accumulator)\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to left\n * to each element with
its index in the original array and current accumulator value.\n * \n * Throws an exception if this array is empty. If
the array can be empty in an expected way,\n * please use [reduceRightIndexedOrNull] instead. It returns `null`
when its receiver is empty.\n * \n * @param [operation] function that takes the index of an element, the element
itself and current accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n */\npublic inline fun <S, T : S> Array<out
T>.reduceRightIndexed(operation: (index: Int, T, acc: S) -> S): S {\n    var index = lastIndex\n    if (index < 0) throw
UnsupportedOperationException("Empty array can't be reduced.")\n    var accumulator: S = get(index--)\n    while
(index >= 0) {\n        accumulator = operation(index, get(index), accumulator)\n        --index\n    }\n    return

```

```

accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to
left\n * to each element with its index in the original array and current accumulator value.\n * \n * Throws an
exception if this array is empty. If the array can be empty in an expected way,\n * please use
[reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n * @param [operation]
function that takes the index of an element, the element itself and current accumulator
value,\n * and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceRight\n */\npublic
inline fun ByteArray.reduceRightIndexed(operation: (index: Int, Byte, acc: Byte) -> Byte): Byte {\n    var index =
lastIndex\n    if (index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n    var
accumulator = get(index--)\n    while (index >= 0) {\n        accumulator = operation(index, get(index),
accumulator)\n        --index\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the last
element and applying [operation] from right to left\n * to each element with its index in the original array and
current accumulator value.\n * \n * Throws an exception if this array is empty. If the array can be empty in an
expected way,\n * please use [reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n
* @param [operation] function that takes the index of an element, the element itself and current accumulator
value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n */\npublic inline fun
ShortArray.reduceRightIndexed(operation: (index: Int, Short, acc: Short) -> Short): Short {\n    var index =
lastIndex\n    if (index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n    var
accumulator = get(index--)\n    while (index >= 0) {\n        accumulator = operation(index, get(index),
accumulator)\n        --index\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the last
element and applying [operation] from right to left\n * to each element with its index in the original array and
current accumulator value.\n * \n * Throws an exception if this array is empty. If the array can be empty in an
expected way,\n * please use [reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n
* @param [operation] function that takes the index of an element, the element itself and current accumulator
value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n */\npublic inline fun
IntArray.reduceRightIndexed(operation: (index: Int, Int, acc: Int) -> Int): Int {\n    var index = lastIndex\n    if (index
< 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n    var accumulator = get(index--)\n
while (index >= 0) {\n        accumulator = operation(index, get(index), accumulator)\n        --index\n    }\n    return
accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to
left\n * to each element with its index in the original array and current accumulator value.\n * \n * Throws an
exception if this array is empty. If the array can be empty in an expected way,\n * please use
[reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n * @param [operation]
function that takes the index of an element, the element itself and current accumulator
value,\n * and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceRight\n */\npublic
inline fun LongArray.reduceRightIndexed(operation: (index: Int, Long, acc: Long) -> Long): Long {\n    var index =
lastIndex\n    if (index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n    var
accumulator = get(index--)\n    while (index >= 0) {\n        accumulator = operation(index, get(index),
accumulator)\n        --index\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the last
element and applying [operation] from right to left\n * to each element with its index in the original array and
current accumulator value.\n * \n * Throws an exception if this array is empty. If the array can be empty in an
expected way,\n * please use [reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n
* @param [operation] function that takes the index of an element, the element itself and current accumulator
value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n */\npublic inline fun
FloatArray.reduceRightIndexed(operation: (index: Int, Float, acc: Float) -> Float): Float {\n    var index =
lastIndex\n    if (index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n    var
accumulator = get(index--)\n    while (index >= 0) {\n        accumulator = operation(index, get(index),

```

```

accumulator)\n    --index\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the last
element and applying [operation] from right to left\n * to each element with its index in the original array and
current accumulator value.\n * \n * Throws an exception if this array is empty. If the array can be empty in an
expected way,\n * please use [reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n
* @param [operation] function that takes the index of an element, the element itself and current accumulator
value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n */\n\npublic inline fun
DoubleArray.reduceRightIndexed(operation: (index: Int, Double, acc: Double) -> Double): Double {\n    var index =
lastIndex\n    if (index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n    var
accumulator = get(index--)\n    while (index >= 0) {\n        accumulator = operation(index, get(index),
accumulator)\n        --index\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the last
element and applying [operation] from right to left\n * to each element with its index in the original array and
current accumulator value.\n * \n * Throws an exception if this array is empty. If the array can be empty in an
expected way,\n * please use [reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n
* @param [operation] function that takes the index of an element, the element itself and current accumulator
value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n */\n\npublic inline fun
BooleanArray.reduceRightIndexed(operation: (index: Int, Boolean, acc: Boolean) -> Boolean): Boolean {\n    var
index = lastIndex\n    if (index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n    var
accumulator = get(index--)\n    while (index >= 0) {\n        accumulator = operation(index, get(index),
accumulator)\n        --index\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the last
element and applying [operation] from right to left\n * to each element with its index in the original array and
current accumulator value.\n * \n * Throws an exception if this array is empty. If the array can be empty in an
expected way,\n * please use [reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n
* @param [operation] function that takes the index of an element, the element itself and current accumulator
value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n */\n\npublic inline fun
CharArray.reduceRightIndexed(operation: (index: Int, Char, acc: Char) -> Char): Char {\n    var index = lastIndex\n
if (index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n    var accumulator =
get(index--)\n    while (index >= 0) {\n        accumulator = operation(index, get(index), accumulator)\n        --index\n
    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation]
from right to left\n * to each element with its index in the original array and current accumulator value.\n * \n
* Returns `null` if the array is empty.\n * \n * @param [operation] function that takes the index of an element, the
element itself and current accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n */\n\n@SinceKotlin("1.4")\n\npublic inline fun <S,
T : S> Array<out T>.reduceRightIndexedOrNull(operation: (index: Int, T, acc: S) -> S): S? {\n    var index =
lastIndex\n    if (index < 0) return null\n    var accumulator: S = get(index--)\n    while (index >= 0) {\n
accumulator = operation(index, get(index), accumulator)\n        --index\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to left\n * to each element with
its index in the original array and current accumulator value.\n * \n * Returns `null` if the array is empty.\n * \n
* @param [operation] function that takes the index of an element, the element itself and current accumulator value,\n
* and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n */\n\n@SinceKotlin("1.4")\n\npublic inline fun
ByteArray.reduceRightIndexedOrNull(operation: (index: Int, Byte, acc: Byte) -> Byte): Byte? {\n    var index =
lastIndex\n    if (index < 0) return null\n    var accumulator = get(index--)\n    while (index >= 0) {\n
accumulator = operation(index, get(index), accumulator)\n        --index\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to left\n * to each element with
its index in the original array and current accumulator value.\n * \n * Returns `null` if the array is empty.\n * \n
* @param [operation] function that takes the index of an element, the element itself and current accumulator value,\n
* and calculates the next accumulator value.\n * \n * @sample

```

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value. @sample

```

samples.collections.Collections.Aggregates.reduceRightOrNull
^
@SinceKotlin("1.4")
public inline fun
ShortArray.reduceRightIndexedOrNull(operation: (index: Int, Short, acc: Short) -> Short): Short? {
    var index = lastIndex
    if (index < 0) return null
    var accumulator = get(index--)
    while (index >= 0) {
        accumulator = operation(index, get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with the last element and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns `null` if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value. @sample

```

samples.collections.Collections.Aggregates.reduceRightOrNull
^
@SinceKotlin("1.4")
public inline fun
IntArray.reduceRightIndexedOrNull(operation: (index: Int, Int, acc: Int) -> Int): Int? {
    var index = lastIndex
    if (index < 0) return null
    var accumulator = get(index--)
    while (index >= 0) {
        accumulator =
operation(index, get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with the last element and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns `null` if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value. @sample

```

samples.collections.Collections.Aggregates.reduceRightOrNull
^
@SinceKotlin("1.4")
public inline fun
LongArray.reduceRightIndexedOrNull(operation: (index: Int, Long, acc: Long) -> Long): Long? {
    var index = lastIndex
    if (index < 0) return null
    var accumulator = get(index--)
    while (index >= 0) {
        accumulator = operation(index, get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with the last element and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns `null` if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value. @sample

```

samples.collections.Collections.Aggregates.reduceRightOrNull
^
@SinceKotlin("1.4")
public inline fun
FloatArray.reduceRightIndexedOrNull(operation: (index: Int, Float, acc: Float) -> Float): Float? {
    var index = lastIndex
    if (index < 0) return null
    var accumulator = get(index--)
    while (index >= 0) {
        accumulator = operation(index, get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with the last element and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns `null` if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value. @sample

```

samples.collections.Collections.Aggregates.reduceRightOrNull
^
@SinceKotlin("1.4")
public inline fun
DoubleArray.reduceRightIndexedOrNull(operation: (index: Int, Double, acc: Double) -> Double): Double? {
    var index = lastIndex
    if (index < 0) return null
    var accumulator = get(index--)
    while (index >= 0) {
        accumulator = operation(index, get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with the last element and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns `null` if the array is empty.

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value. @sample

```

samples.collections.Collections.Aggregates.reduceRightOrNull
^
@SinceKotlin("1.4")
public inline fun
BooleanArray.reduceRightIndexedOrNull(operation: (index: Int, Boolean, acc: Boolean) -> Boolean): Boolean? {
    var index = lastIndex
    if (index < 0) return null
    var accumulator = get(index--)
    while (index >= 0) {
        accumulator = operation(index, get(index), accumulator)
        --index
    }
    return accumulator
}

```

Accumulates value starting with the last element and applying [operation] from right to left to each element with its index in the original array and current accumulator value. Returns `null` if the array is empty.

```

@param [operation] function that takes the index of an element, the element itself and current accumulator value,\n
* and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n * \n *\n@SinceKotlin("1.4")\npublic inline fun
CharArray.reduceRightIndexedOrNull(operation: (index: Int, Char, acc: Char) -> Char): Char? {\n  var index =
lastIndex\n  if (index < 0) return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n
accumulator = operation(index, get(index), accumulator)\n  --index\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to left\n * to each element and
current accumulator value.\n * \n * Returns `null` if the array is empty.\n * \n * @param [operation] function that
takes an element and current accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun <S, T : S>
Array<out T>.reduceRightOrNull(operation: (T, acc: S) -> S): S? {\n  var index = lastIndex\n  if (index < 0)
return null\n  var accumulator: S = get(index--)\n  while (index >= 0) {\n    accumulator = operation(get(index--),
accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with the last element and
applying [operation] from right to left\n * to each element and current accumulator value.\n * \n * Returns `null` if
the array is empty.\n * \n * @param [operation] function that takes an element and current accumulator value,\n *
and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
ByteArray.reduceRightOrNull(operation: (Byte, acc: Byte) -> Byte): Byte? {\n  var index = lastIndex\n  if (index
< 0) return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n    accumulator =
operation(get(index--), accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with the
last element and applying [operation] from right to left\n * to each element and current accumulator value.\n * \n *
Returns `null` if the array is empty.\n * \n * @param [operation] function that takes an element and current
accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
ShortArray.reduceRightOrNull(operation: (Short, acc: Short) -> Short): Short? {\n  var index = lastIndex\n  if
(index < 0) return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n    accumulator =
operation(get(index--), accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with the
last element and applying [operation] from right to left\n * to each element and current accumulator value.\n * \n *
Returns `null` if the array is empty.\n * \n * @param [operation] function that takes an element and current
accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
IntArray.reduceRightOrNull(operation: (Int, acc: Int) -> Int): Int? {\n  var index = lastIndex\n  if (index < 0)
return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n    accumulator = operation(get(index--),
accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with the last element and
applying [operation] from right to left\n * to each element and current accumulator value.\n * \n * Returns `null` if
the array is empty.\n * \n * @param [operation] function that takes an element and current accumulator value,\n *
and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
LongArray.reduceRightOrNull(operation: (Long, acc: Long) -> Long): Long? {\n  var index = lastIndex\n  if
(index < 0) return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n    accumulator =
operation(get(index--), accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with the
last element and applying [operation] from right to left\n * to each element and current accumulator value.\n * \n *
Returns `null` if the array is empty.\n * \n * @param [operation] function that takes an element and current

```

```

accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
FloatArray.reduceRightOrNull(operation: (Float, acc: Float) -> Float): Float? {\n  var index = lastIndex\n  if
(index < 0) return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n    accumulator =
operation(get(index--), accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with the
last element and applying [operation] from right to left\n * to each element and current accumulator value.\n * \n *
Returns `null` if the array is empty.\n * \n * @param [operation] function that takes an element and current
accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
DoubleArray.reduceRightOrNull(operation: (Double, acc: Double) -> Double): Double? {\n  var index =
lastIndex\n  if (index < 0) return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n
accumulator = operation(get(index--), accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value
starting with the last element and applying [operation] from right to left\n * to each element and current accumulator
value.\n * \n * Returns `null` if the array is empty.\n * \n * @param [operation] function that takes an element and
current accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
BooleanArray.reduceRightOrNull(operation: (Boolean, acc: Boolean) -> Boolean): Boolean? {\n  var index =
lastIndex\n  if (index < 0) return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n
accumulator = operation(get(index--), accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value
starting with the last element and applying [operation] from right to left\n * to each element and current accumulator
value.\n * \n * Returns `null` if the array is empty.\n * \n * @param [operation] function that takes an element and
current accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
CharArray.reduceRightOrNull(operation: (Char, acc: Char) -> Char): Char? {\n  var index = lastIndex\n  if (index
< 0) return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n    accumulator =
operation(get(index--), accumulator)\n  }\n  return accumulator\n}\n\n/**\n * Returns a list containing successive
accumulation values generated by applying [operation] from left to right\n * to each element and current
accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should
not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation]
function that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n *
@sample samples.collections.Collections.Aggregates.runningFold\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
<T, R> Array<out T>.runningFold(initial: R, operation: (acc: R, T) -> R): List<R> {\n  if (isEmpty()) return
listOf(initial)\n  val result = ArrayList<R>(size + 1).apply { add(initial) }\n  var accumulator = initial\n  for
(element in this) {\n    accumulator = operation(accumulator, element)\n    result.add(accumulator)\n  }\n  return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying
[operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n
* Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the
previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an
element, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningFold\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <R> ByteArray.runningFold(initial: R,
operation: (acc: R, Byte) -> R): List<R> {\n  if (isEmpty()) return listOf(initial)\n  val result = ArrayList<R>(size
+ 1).apply { add(initial) }\n  var accumulator = initial\n  for (element in this) {\n    accumulator =
operation(accumulator, element)\n    result.add(accumulator)\n  }\n  return result\n}\n\n/**\n * Returns a list

```

containing successive accumulation values generated by applying [operation] from left to right\n \* to each element and current accumulator value that starts with [initial] value.\n \* \n \* Note that `acc` value passed to [operation] function should not be mutated;\n \* otherwise it would affect the previous value in resulting list.\n \* \n \* @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.\n \* \n \* @sample samples.collections.Collections.Aggregates.runningFold

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <R> ShortArray.runningFold(initial: R, operation: (acc: R, Short) -> R): List<R> {\n    if (isEmpty()) return listOf(initial)\n    val result = ArrayList<R>(size + 1).apply { add(initial) }\n    var accumulator = initial\n    for (element in this) {\n        accumulator = operation(accumulator, element)\n        result.add(accumulator)\n    }\n    return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningFold
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <R> IntArray.runningFold(initial: R, operation: (acc: R, Int) -> R): List<R> {\n    if (isEmpty()) return listOf(initial)\n    val result = ArrayList<R>(size + 1).apply { add(initial) }\n    var accumulator = initial\n    for (element in this) {\n        accumulator = operation(accumulator, element)\n        result.add(accumulator)\n    }\n    return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningFold
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <R> LongArray.runningFold(initial: R, operation: (acc: R, Long) -> R): List<R> {\n    if (isEmpty()) return listOf(initial)\n    val result = ArrayList<R>(size + 1).apply { add(initial) }\n    var accumulator = initial\n    for (element in this) {\n        accumulator = operation(accumulator, element)\n        result.add(accumulator)\n    }\n    return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningFold
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <R> FloatArray.runningFold(initial: R, operation: (acc: R, Float) -> R): List<R> {\n    if (isEmpty()) return listOf(initial)\n    val result = ArrayList<R>(size + 1).apply { add(initial) }\n    var accumulator = initial\n    for (element in this) {\n        accumulator = operation(accumulator, element)\n        result.add(accumulator)\n    }\n    return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningFold
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <R> DoubleArray.runningFold(initial: R, operation: (acc: R, Double) -> R): List<R> {\n    if (isEmpty()) return listOf(initial)\n    val result = ArrayList<R>(size + 1).apply { add(initial) }\n    var accumulator = initial\n    for (element in this) {\n        accumulator = operation(accumulator, element)\n        result.add(accumulator)\n    }\n    return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningFold
```

@param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.runningFold
*/
@SinceKotlin("1.4")
@kotlin.internal.InlineOnly
public inline fun <R> BooleanArray.runningFold(initial: R, operation: (acc: R, Boolean) -> R): List<R> {
    if (isEmpty()) return listOf(initial)
    val result = ArrayList<R>(size + 1).apply { add(initial) }
    var accumulator = initial
    for (element in this) {
        accumulator = operation(accumulator, element)
        result.add(accumulator)
    }
    return result
}

```

Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with [initial] value.

Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.

@param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.runningFold
*/
@SinceKotlin("1.4")
@kotlin.internal.InlineOnly
public inline fun <R> CharArray.runningFold(initial: R, operation: (acc: R, Char) -> R): List<R> {
    if (isEmpty()) return listOf(initial)
    val result = ArrayList<R>(size + 1).apply { add(initial) }
    var accumulator = initial
    for (element in this) {
        accumulator = operation(accumulator, element)
        result.add(accumulator)
    }
    return result
}

```

Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original array and current accumulator value that starts with [initial] value.

Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.

@param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.runningFold
*/
@SinceKotlin("1.4")
public inline fun <T, R> Array<out T>.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, T) -> R): List<R> {
    if (isEmpty()) return listOf(initial)
    val result = ArrayList<R>(size + 1).apply { add(initial) }
    var accumulator = initial
    for (index in indices) {
        accumulator = operation(index, accumulator, this[index])
        result.add(accumulator)
    }
    return result
}

```

Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original array and current accumulator value that starts with [initial] value.

Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.

@param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.runningFold
*/
@SinceKotlin("1.4")
@kotlin.internal.InlineOnly
public inline fun <R> ByteArray.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, Byte) -> R): List<R> {
    if (isEmpty()) return listOf(initial)
    val result = ArrayList<R>(size + 1).apply { add(initial) }
    var accumulator = initial
    for (index in indices) {
        accumulator = operation(index, accumulator, this[index])
        result.add(accumulator)
    }
    return result
}

```

Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original array and current accumulator value that starts with [initial] value.

Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.

@param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.runningFold
*/
@SinceKotlin("1.4")
@kotlin.internal.InlineOnly
public inline fun <R> ShortArray.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, Short) -> R): List<R> {
    if (isEmpty()) return listOf(initial)
    val result = ArrayList<R>(size + 1).apply { add(initial) }
    var accumulator = initial
    for (index in indices) {
        accumulator = operation(index, accumulator, this[index])
        result.add(accumulator)
    }
    return result
}

```

Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original array and current accumulator value that starts with [initial] value.

Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.

@param [operation]

function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

```

@sample samples.collections.Collections.Aggregates.runningFold
*/n@SinceKotlin("1.4")n@kotlin.internal.InlineOnlynpublic inline fun <R>
IntArray.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, Int) -> R): List<R> {n if (isEmpty()) return
listOf(initial)n val result = ArrayList<R>(size + 1).apply { add(initial) }n var accumulator = initialn for
(index in indices) {n accumulator = operation(index, accumulator, this[index])n result.add(accumulator)n
}n return resultn}n/n/**n * Returns a list containing successive accumulation values generated by applying
[operation] from left to rightn * to each element, its index in the original array and current accumulator value that
starts with [initial] value.n * n * Note that `acc` value passed to [operation] function should not be mutated;n *
otherwise it would affect the previous value in resulting list.n * n * @param [operation] function that takes the
index of an element, current accumulator value and the element itself, and calculates the next accumulator
value.n * n * @sample samples.collections.Collections.Aggregates.runningFold
*/n@SinceKotlin("1.4")n@kotlin.internal.InlineOnlynpublic inline fun <R>
LongArray.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, Long) -> R): List<R> {n if (isEmpty())
return listOf(initial)n val result = ArrayList<R>(size + 1).apply { add(initial) }n var accumulator = initialn
for (index in indices) {n accumulator = operation(index, accumulator, this[index])n
result.add(accumulator)n }n return resultn}n/n/**n * Returns a list containing successive accumulation
values generated by applying [operation] from left to rightn * to each element, its index in the original array and
current accumulator value that starts with [initial] value.n * n * Note that `acc` value passed to [operation] function
should not be mutated;n * otherwise it would affect the previous value in resulting list.n * n * @param [operation]
function that takes the index of an element, current accumulator value and the element itself, and calculates the
next accumulator value.n * n * @sample samples.collections.Collections.Aggregates.runningFold
*/n@SinceKotlin("1.4")n@kotlin.internal.InlineOnlynpublic inline fun <R>
FloatArray.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, Float) -> R): List<R> {n if (isEmpty())
return listOf(initial)n val result = ArrayList<R>(size + 1).apply { add(initial) }n var accumulator = initialn
for (index in indices) {n accumulator = operation(index, accumulator, this[index])n
result.add(accumulator)n }n return resultn}n/n/**n * Returns a list containing successive accumulation
values generated by applying [operation] from left to rightn * to each element, its index in the original array and
current accumulator value that starts with [initial] value.n * n * Note that `acc` value passed to [operation] function
should not be mutated;n * otherwise it would affect the previous value in resulting list.n * n * @param [operation]
function that takes the index of an element, current accumulator value and the element itself, and calculates the
next accumulator value.n * n * @sample samples.collections.Collections.Aggregates.runningFold
*/n@SinceKotlin("1.4")n@kotlin.internal.InlineOnlynpublic inline fun <R>
DoubleArray.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, Double) -> R): List<R> {n if
(isEmpty()) return listOf(initial)n val result = ArrayList<R>(size + 1).apply { add(initial) }n var accumulator =
initialn for (index in indices) {n accumulator = operation(index, accumulator, this[index])n
result.add(accumulator)n }n return resultn}n/n/**n * Returns a list containing successive accumulation
values generated by applying [operation] from left to rightn * to each element, its index in the original array and
current accumulator value that starts with [initial] value.n * n * Note that `acc` value passed to [operation] function
should not be mutated;n * otherwise it would affect the previous value in resulting list.n * n * @param [operation]
function that takes the index of an element, current accumulator value and the element itself, and calculates the
next accumulator value.n * n * @sample samples.collections.Collections.Aggregates.runningFold
*/n@SinceKotlin("1.4")n@kotlin.internal.InlineOnlynpublic inline fun <R>
BooleanArray.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, Boolean) -> R): List<R> {n if
(isEmpty()) return listOf(initial)n val result = ArrayList<R>(size + 1).apply { add(initial) }n var accumulator =
initialn for (index in indices) {n accumulator = operation(index, accumulator, this[index])n
result.add(accumulator)n }n return resultn}n/n/**n * Returns a list containing successive accumulation
values generated by applying [operation] from left to rightn * to each element, its index in the original array and

```

current accumulator value that starts with [initial] value.\n \* \n \* Note that `acc` value passed to [operation] function should not be mutated;\n \* otherwise it would affect the previous value in resulting list.\n \* \n \* @param [operation] function that takes the index of an element, current accumulator value\n \* and the element itself, and calculates the next accumulator value.\n \* \n \* @sample samples.collections.Collections.Aggregates.runningFold\n

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <R>
CharArray.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, Char) -> R): List<R> {\n if (isEmpty())
return listOf(initial)\n val result = ArrayList<R>(size + 1).apply { add(initial) }\n var accumulator = initial\n for (index in indices) {\n accumulator = operation(index, accumulator, this[index])\n result.add(accumulator)\n }\n return result\n}\n\n/**\n * Returns a list containing successive accumulation
values generated by applying [operation] from left to right\n * to each element and current accumulator value that
starts with the first element of this array.\n * \n * Note that `acc` value passed to [operation] function should not be
mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that
takes current accumulator value and the element, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningReduce\n
```

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun <S, T : S>
Array<out T>.runningReduce(operation: (acc: S, T) -> S): List<S> {\n if (isEmpty()) return emptyList()\n var
accumulator: S = this[0]\n val result = ArrayList<S>(size).apply { add(accumulator) }\n for (index in 1 until
size) {\n accumulator = operation(accumulator, this[index])\n result.add(accumulator)\n }\n return
result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from
left to right\n * to each element and current accumulator value that starts with the first element of this array.\n * \n *
@param [operation] function that takes current accumulator value and an element, and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningReduce\n
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.runningReduce(operation:
(acc: Byte, Byte) -> Byte): List<Byte> {\n if (isEmpty()) return emptyList()\n var accumulator = this[0]\n val
result = ArrayList<Byte>(size).apply { add(accumulator) }\n for (index in 1 until size) {\n accumulator =
operation(accumulator, this[index])\n result.add(accumulator)\n }\n return result\n}\n\n/**\n * Returns a list
containing successive accumulation values generated by applying [operation] from left to right\n * to each element
and current accumulator value that starts with the first element of this array.\n * \n * @param [operation] function
that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningReduce\n
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.runningReduce(operation:
(acc: Short, Short) -> Short): List<Short> {\n if (isEmpty()) return emptyList()\n var accumulator = this[0]\n
val result = ArrayList<Short>(size).apply { add(accumulator) }\n for (index in 1 until size) {\n accumulator =
operation(accumulator, this[index])\n result.add(accumulator)\n }\n return result\n}\n\n/**\n * Returns a list
containing successive accumulation values generated by applying [operation] from left to right\n * to each element
and current accumulator value that starts with the first element of this array.\n * \n * @param [operation] function
that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningReduce\n
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.runningReduce(operation: (acc:
Int, Int) -> Int): List<Int> {\n if (isEmpty()) return emptyList()\n var accumulator = this[0]\n val result =
ArrayList<Int>(size).apply { add(accumulator) }\n for (index in 1 until size) {\n accumulator =
operation(accumulator, this[index])\n result.add(accumulator)\n }\n return result\n}\n\n/**\n * Returns a list
containing successive accumulation values generated by applying [operation] from left to right\n * to each element
and current accumulator value that starts with the first element of this array.\n * \n * @param [operation] function
that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningReduce\n
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.runningReduce(operation:
(acc: Long, Long) -> Long): List<Long> {\n if (isEmpty()) return emptyList()\n var accumulator = this[0]\n
```

```
val result = ArrayList<Long>(size).apply { add(accumulator) }
for (index in 1 until size) {
    accumulator = operation(accumulator, this[index])
    result.add(accumulator)
}
return result
```

\* Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with the first element of this array.

@param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

@sample samples.collections.Collections.Aggregates.runningReduce

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.runningReduce(operation: (acc: Float, Float) -> Float): List<Float> {
    if (isEmpty()) return emptyList()
    var accumulator = this[0]
    val result = ArrayList<Float>(size).apply { add(accumulator) }
    for (index in 1 until size) {
        accumulator = operation(accumulator, this[index])
        result.add(accumulator)
    }
    return result
}
```

\* Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with the first element of this array.

@param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

@sample samples.collections.Collections.Aggregates.runningReduce

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.runningReduce(operation: (acc: Double, Double) -> Double): List<Double> {
    if (isEmpty()) return emptyList()
    var accumulator = this[0]
    val result = ArrayList<Double>(size).apply { add(accumulator) }
    for (index in 1 until size) {
        accumulator = operation(accumulator, this[index])
        result.add(accumulator)
    }
    return result
}
```

\* Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with the first element of this array.

@param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

@sample samples.collections.Collections.Aggregates.runningReduce

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.runningReduce(operation: (acc: Boolean, Boolean) -> Boolean): List<Boolean> {
    if (isEmpty()) return emptyList()
    var accumulator = this[0]
    val result = ArrayList<Boolean>(size).apply { add(accumulator) }
    for (index in 1 until size) {
        accumulator = operation(accumulator, this[index])
        result.add(accumulator)
    }
    return result
}
```

\* Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with the first element of this array.

@param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.

@sample samples.collections.Collections.Aggregates.runningReduce

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.runningReduce(operation: (acc: Char, Char) -> Char): List<Char> {
    if (isEmpty()) return emptyList()
    var accumulator = this[0]
    val result = ArrayList<Char>(size).apply { add(accumulator) }
    for (index in 1 until size) {
        accumulator = operation(accumulator, this[index])
        result.add(accumulator)
    }
    return result
}
```

\* Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original array and current accumulator value that starts with the first element of this array.

Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.

@param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

@sample samples.collections.Collections.Aggregates.runningReduce

```
*\n@SinceKotlin("1.4")\npublic inline fun <S, T : S> Array<out T>.runningReduceIndexed(operation: (index: Int, acc: S, T) -> S): List<S> {
    if (isEmpty()) return emptyList()
    var accumulator: S = this[0]
    val result = ArrayList<S>(size).apply { add(accumulator) }
    for (index in 1 until size) {
        accumulator = operation(index, accumulator, this[index])
        result.add(accumulator)
    }
    return result
}
```

\* Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original array and current accumulator value that starts with the first element of this array.

@param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.

@sample samples.collections.Collections.Aggregates.runningReduce

```

*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun
ByteArray.runningReduceIndexed(operation: (index: Int, acc: Byte, Byte) -> Byte): List<Byte> {\n  if (isEmpty())
return emptyList()\n  var accumulator = this[0]\n  val result = ArrayList<Byte>(size).apply { add(accumulator)
}\n  for (index in 1 until size) {\n    accumulator = operation(index, accumulator, this[index])\n
result.add(accumulator)\n  }\n  return result\n}\n\n/**\n * Returns a list containing successive accumulation
values generated by applying [operation] from left to right\n * to each element, its index in the original array and
current accumulator value that starts with the first element of this array.\n * \n * @param [operation] function that
takes the index of an element, current accumulator value\n * and the element itself, and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningReduce\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun
ShortArray.runningReduceIndexed(operation: (index: Int, acc: Short, Short) -> Short): List<Short> {\n  if
(isEmpty()) return emptyList()\n  var accumulator = this[0]\n  val result = ArrayList<Short>(size).apply {
add(accumulator) }\n  for (index in 1 until size) {\n    accumulator = operation(index, accumulator, this[index])\n
result.add(accumulator)\n  }\n  return result\n}\n\n/**\n * Returns a list containing successive accumulation
values generated by applying [operation] from left to right\n * to each element, its index in the original array and
current accumulator value that starts with the first element of this array.\n * \n * @param [operation] function that
takes the index of an element, current accumulator value\n * and the element itself, and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningReduce\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun
IntArray.runningReduceIndexed(operation: (index: Int, acc: Int, Int) -> Int): List<Int> {\n  if (isEmpty()) return
emptyList()\n  var accumulator = this[0]\n  val result = ArrayList<Int>(size).apply { add(accumulator) }\n  for
(index in 1 until size) {\n    accumulator = operation(index, accumulator, this[index])\n
result.add(accumulator)\n  }\n  return result\n}\n\n/**\n * Returns a list containing successive accumulation
values generated by applying [operation] from left to right\n * to each element, its index in the original array and
current accumulator value that starts with the first element of this array.\n * \n * @param [operation] function that
takes the index of an element, current accumulator value\n * and the element itself, and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningReduce\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun
LongArray.runningReduceIndexed(operation: (index: Int, acc: Long, Long) -> Long): List<Long> {\n  if
(isEmpty()) return emptyList()\n  var accumulator = this[0]\n  val result = ArrayList<Long>(size).apply {
add(accumulator) }\n  for (index in 1 until size) {\n    accumulator = operation(index, accumulator, this[index])\n
result.add(accumulator)\n  }\n  return result\n}\n\n/**\n * Returns a list containing successive accumulation
values generated by applying [operation] from left to right\n * to each element, its index in the original array and
current accumulator value that starts with the first element of this array.\n * \n * @param [operation] function that
takes the index of an element, current accumulator value\n * and the element itself, and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningReduce\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun
FloatArray.runningReduceIndexed(operation: (index: Int, acc: Float, Float) -> Float): List<Float> {\n  if
(isEmpty()) return emptyList()\n  var accumulator = this[0]\n  val result = ArrayList<Float>(size).apply {
add(accumulator) }\n  for (index in 1 until size) {\n    accumulator = operation(index, accumulator, this[index])\n
result.add(accumulator)\n  }\n  return result\n}\n\n/**\n * Returns a list containing successive accumulation
values generated by applying [operation] from left to right\n * to each element, its index in the original array and
current accumulator value that starts with the first element of this array.\n * \n * @param [operation] function that
takes the index of an element, current accumulator value\n * and the element itself, and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningReduce\n
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun
DoubleArray.runningReduceIndexed(operation: (index: Int, acc: Double, Double) -> Double): List<Double> {\n  if
(isEmpty()) return emptyList()\n  var accumulator = this[0]\n  val result = ArrayList<Double>(size).apply {

```

```

add(accumulator) }\n for (index in 1 until size) {\n accumulator = operation(index, accumulator, this[index])\n
result.add(accumulator)\n }\n return result\n}\n\n/**\n * Returns a list containing successive accumulation
values generated by applying [operation] from left to right\n * to each element, its index in the original array and
current accumulator value that starts with the first element of this array.\n * \n * @param [operation] function that
takes the index of an element, current accumulator value\n * and the element itself, and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningReduce\n
*/\n\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun
BooleanArray.runningReduceIndexed(operation: (index: Int, acc: Boolean, Boolean) -> Boolean): List<Boolean>
{\n if (isEmpty()) return emptyList()\n var accumulator = this[0]\n val result =
ArrayList<Boolean>(size).apply { add(accumulator) }\n for (index in 1 until size) {\n accumulator =
operation(index, accumulator, this[index])\n result.add(accumulator)\n }\n return result\n}\n\n/**\n *
Returns a list containing successive accumulation values generated by applying [operation] from left to right\n *
to each element, its index in the original array and current accumulator value that starts with the first element of
this
array.\n * \n * @param [operation] function that takes the index of an element, current accumulator value\n * and
the element itself, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningReduce\n
*/\n\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun
CharArray.runningReduceIndexed(operation: (index: Int, acc: Char, Char) -> Char): List<Char> {\n if (isEmpty())
return emptyList()\n var accumulator = this[0]\n val result = ArrayList<Char>(size).apply { add(accumulator)
}\n for (index in 1 until size) {\n accumulator = operation(index, accumulator, this[index])\n
result.add(accumulator)\n }\n return result\n}\n\n/**\n * Returns a list containing successive accumulation
values generated by applying [operation] from left to right\n * to each element and current accumulator value that
starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n *
otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes
current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.scan\n
*/\n\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun <T, R>
Array<out T>.scan(initial: R, operation: (acc: R, T) -> R): List<R> {\n return runningFold(initial,
operation)\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation]
from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n *
Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the
previous value in
resulting list.\n * \n * @param [operation] function that takes current accumulator value and an element, and
calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*/\n\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun <R> ByteArray.scan(initial: R, operation: (acc: R, Byte) -> R): List<R> {\n return runningFold(initial,
operation)\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation]
from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n *
Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the
previous value in
resulting list.\n * \n * @param [operation] function that takes current accumulator value and an element, and
calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*/\n\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun <R> ShortArray.scan(initial: R, operation: (acc: R, Short) -> R): List<R> {\n return
runningFold(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values generated by
applying [operation] from left to right\n * to each element and current accumulator value that starts with [initial]
value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would
affect the previous value in
resulting list.\n * \n * @param [operation] function that takes current accumulator value
and an element, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.scan\n

```

```

*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun <R> IntArray.scan(initial: R, operation: (acc: R, Int) -> R): List<R> {\n    return runningFold(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun <R> LongArray.scan(initial: R, operation: (acc: R, Long) -> R): List<R> {\n    return runningFold(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun <R> FloatArray.scan(initial: R, operation: (acc: R, Float) -> R): List<R> {\n    return runningFold(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun <R> DoubleArray.scan(initial: R, operation: (acc: R, Double) -> R): List<R> {\n    return runningFold(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun <R> BooleanArray.scan(initial: R, operation: (acc: R, Boolean) -> R): List<R> {\n    return runningFold(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun <R> CharArray.scan(initial: R, operation: (acc: R, Char) -> R): List<R> {\n    return runningFold(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from left to right\n * to each element, its index in the original array and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes the index of an element, current accumulator value\n * and the element itself, and calculates the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun <T, R>

```

```

Array<out T>.scanIndexed(initial: R, operation: (index: Int, acc: R, T) -> R): List<R> {\n  return
runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values
generated by applying [operation] from left to right\n * to each element, its index in the original array and current
accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should
not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation]
function that takes the index of an element, current accumulator value\n * and the element itself, and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun <R> ByteArray.scanIndexed(initial: R, operation: (index: Int, acc: R, Byte) -> R): List<R> {\n  return
runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values
generated by applying [operation] from left to right\n * to each element, its index in the original array and current
accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should
not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation]
function that takes the index of an element, current accumulator value\n * and the element itself, and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun <R> ShortArray.scanIndexed(initial: R, operation: (index: Int, acc: R, Short) -> R): List<R> {\n  return
runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values
generated by applying [operation] from left to right\n * to each element, its index in the original array and current
accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should
not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation]
function that takes the index of an element, current accumulator value\n * and the element itself, and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun <R> IntArray.scanIndexed(initial: R, operation: (index: Int, acc: R, Int) -> R): List<R> {\n  return
runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values
generated by applying [operation] from left to right\n * to each element, its index in the original array and current
accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should
not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation]
function that takes the index of an element, current accumulator value\n * and the element itself, and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun <R> LongArray.scanIndexed(initial: R, operation: (index: Int, acc: R, Long) -> R): List<R> {\n  return
runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values
generated by applying [operation] from left to right\n * to each element, its index in the original array and current
accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should
not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation]
function that takes the index of an element, current accumulator value\n * and the element itself, and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun <R> FloatArray.scanIndexed(initial: R, operation: (index: Int, acc: R, Float) -> R): List<R> {\n  return
runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values
generated by applying [operation] from left to right\n * to each element, its index in the original array and current
accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should
not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation]
function that takes the index of an element, current accumulator value\n * and the element itself, and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic

```

```

c inline fun <R> DoubleArray.scanIndexed(initial: R, operation: (index: Int, acc: R, Double) -> R): List<R> {\n
return runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values
generated by applying [operation] from left to right\n * to each element, its index in the original array and current
accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should
not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation]
function that takes the index of an element, current accumulator value\n * and the element itself, and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*/\n\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
c inline fun <R> BooleanArray.scanIndexed(initial: R, operation: (index: Int, acc: R, Boolean) -> R): List<R> {\n
return runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation values
generated by applying [operation] from left to right\n * to each element, its index in the original array and current
accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should
not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation]
function that takes the index of an element, current accumulator value\n * and the element itself, and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*/\n\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
c inline fun <R> CharArray.scanIndexed(initial: R, operation: (index: Int, acc: R, Char) -> R): List<R> {\n
return runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n */\n\n@Deprecated("Use sumOf instead.")\n
ReplaceWith("this.sumOf(selector)")\n\n@DeprecatedSinceKotlin(warningSince = "1.5")\n\npublic inline fun <T>
Array<out T>.sumBy(selector: (T) -> Int): Int {\n
var sum: Int = 0\n
for (element in this) {\n
sum +=
selector(element)\n
}\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n */\n\n@Deprecated("Use sumOf instead.")\n
ReplaceWith("this.sumOf(selector)")\n\n@DeprecatedSinceKotlin(warningSince = "1.5")\n\npublic inline fun
ByteArray.sumBy(selector: (Byte) -> Int): Int {\n
var sum: Int = 0\n
for (element in this) {\n
sum +=
selector(element)\n
}\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n */\n\n@Deprecated("Use sumOf instead.")\n
ReplaceWith("this.sumOf(selector)")\n\n@DeprecatedSinceKotlin(warningSince = "1.5")\n\npublic inline fun
ShortArray.sumBy(selector: (Short) -> Int): Int {\n
var sum: Int = 0\n
for (element in this) {\n
sum +=
selector(element)\n
}\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n */\n\n@Deprecated("Use sumOf instead.")\n
ReplaceWith("this.sumOf(selector)")\n\n@DeprecatedSinceKotlin(warningSince = "1.5")\n\npublic inline fun
IntArray.sumBy(selector: (Int) -> Int): Int {\n
var sum: Int = 0\n
for (element in this) {\n
sum +=
selector(element)\n
}\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n */\n\n@Deprecated("Use sumOf instead.")\n
ReplaceWith("this.sumOf(selector)")\n\n@DeprecatedSinceKotlin(warningSince = "1.5")\n\npublic inline fun
LongArray.sumBy(selector: (Long) -> Int): Int {\n
var sum: Int = 0\n
for (element in this) {\n
sum +=
selector(element)\n
}\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n */\n\n@Deprecated("Use sumOf instead.")\n
ReplaceWith("this.sumOf(selector)")\n\n@DeprecatedSinceKotlin(warningSince = "1.5")\n\npublic inline fun
FloatArray.sumBy(selector: (Float) -> Int): Int {\n
var sum: Int = 0\n
for (element in this) {\n
sum +=
selector(element)\n
}\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n */\n\n@Deprecated("Use sumOf instead.")\n
ReplaceWith("this.sumOf(selector)")\n\n@DeprecatedSinceKotlin(warningSince = "1.5")\n\npublic inline fun
DoubleArray.sumBy(selector: (Double) -> Int): Int {\n
var sum: Int = 0\n
for (element in this) {\n
sum +=
selector(element)\n
}\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n */\n\n@Deprecated("Use sumOf instead.")\n
ReplaceWith("this.sumOf(selector)")\n\n@DeprecatedSinceKotlin(warningSince = "1.5")\n\npublic inline fun

```

```

BooleanArray.sumBy(selector: (Boolean) -> Int): Int {\n  var sum: Int = 0\n  for (element in this) {\n    sum +=
selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n */\n@Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun
CharArray.sumBy(selector: (Char) -> Int): Int {\n  var sum: Int = 0\n  for (element in this) {\n    sum +=
selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n */\n@Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun <T>
Array<out T>.sumByDouble(selector: (T) -> Double): Double {\n  var sum: Double = 0.0\n  for (element in this)
{\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the array.\n */\n@Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun
ByteArray.sumByDouble(selector: (Byte) -> Double): Double {\n  var sum: Double = 0.0\n  for (element in this)
{\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the array.\n */\n@Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun
ShortArray.sumByDouble(selector: (Short) -> Double): Double {\n  var sum: Double = 0.0\n  for (element in this)
{\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the array.\n */\n@Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun
IntArray.sumByDouble(selector: (Int) -> Double): Double {\n  var sum: Double = 0.0\n  for (element in this) {\n
sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the array.\n */\n@Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun
LongArray.sumByDouble(selector: (Long) -> Double): Double {\n  var sum: Double = 0.0\n  for (element in this)
{\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the array.\n */\n@Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun
FloatArray.sumByDouble(selector: (Float) -> Double): Double {\n  var sum: Double = 0.0\n  for (element in this)
{\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the array.\n */\n@Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun
DoubleArray.sumByDouble(selector: (Double) -> Double): Double {\n  var sum: Double = 0.0\n  for (element in
this) {\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the array.\n */\n@Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun
BooleanArray.sumByDouble(selector: (Boolean) -> Double): Double {\n  var sum: Double = 0.0\n  for (element
in this) {\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced
by [selector] function applied to each element in the array.\n */\n@Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun
CharArray.sumByDouble(selector: (Char) -> Double): Double {\n  var sum: Double = 0.0\n  for (element in this)
{\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the array.\n */\n@Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun
*@\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfDouble")\n@kotlin.internal.InlineOnly\npublic inline fun
<T> Array<out T>.sumOf(selector: (T) -> Double): Double {\n  var sum: Double = 0.toDouble()\n  for (element
in this) {\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced
by [selector] function applied to each element in the array.\n */

```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfDouble")\n@kotlin.internal.InlineOnly\npublic inline fun\nByteArray.sumOf(selector: (Byte) -> Double): Double {\n    var sum: Double = 0.toDouble()\n    for (element in\n    this) {\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by\n [selector] function applied to each element in the array.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfDouble")\n@kotlin.internal.InlineOnly\npublic inline fun\nShortArray.sumOf(selector: (Short) -> Double): Double {\n    var sum: Double = 0.toDouble()\n    for (element in\n    this) {\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by\n [selector] function applied to each element in the array.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfDouble")\n@kotlin.internal.InlineOnly\npublic inline fun\nIntArray.sumOf(selector: (Int) -> Double): Double {\n    var sum: Double = 0.toDouble()\n    for (element in this)\n    {\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by\n [selector] function applied to each element in the array.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfDouble")\n@kotlin.internal.InlineOnly\npublic inline fun\nLongArray.sumOf(selector: (Long) -> Double): Double {\n    var sum: Double = 0.toDouble()\n    for (element in\n    this) {\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by\n [selector] function applied to each element in the array.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfDouble")\n@kotlin.internal.InlineOnly\npublic inline fun\nFloatArray.sumOf(selector: (Float) -> Double): Double {\n    var sum: Double = 0.toDouble()\n    for (element in\n    this) {\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by\n [selector] function applied to each element in the array.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfDouble")\n@kotlin.internal.InlineOnly\npublic inline fun\nDoubleArray.sumOf(selector: (Double) -> Double): Double {\n    var sum: Double = 0.toDouble()\n    for (element\n    in this) {\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced\n by [selector] function applied to each element in the array.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfDouble")\n@kotlin.internal.InlineOnly\npublic inline fun\nBooleanArray.sumOf(selector: (Boolean) -> Double): Double {\n    var sum: Double = 0.toDouble()\n    for\n    (element in this) {\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values\n produced by [selector] function applied to each element in the array.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfDouble")\n@kotlin.internal.InlineOnly\npublic inline fun\nCharArray.sumOf(selector: (Char) -> Double): Double {\n    var sum: Double = 0.toDouble()\n    for (element in\n    this) {\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by\n [selector] function applied to each element in the array.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfInt")\n@kotlin.internal.InlineOnly\npublic inline fun <T>\nArray<out T>.sumOf(selector: (T) -> Int): Int {\n    var sum: Int = 0.toInt()\n    for (element in this) {\n        sum +=\n        selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function\n applied to each element in the array.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfInt")\n@kotlin.internal.InlineOnly\npublic inline fun\nByteArray.sumOf(selector: (Byte) -> Int): Int {\n    var sum: Int = 0.toInt()\n    for (element in this) {\n        sum +=\n
```

```

selector(element)\n } \n return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfInt")\n@kotlin.internal.InlineOnly\npublic inline fun
ShortArray.sumOf(selector: (Short) -> Int): Int {\n var sum: Int = 0.toInt()\n for (element in this) {\n sum +=
selector(element)\n } \n return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfInt")\n@kotlin.internal.InlineOnly\npublic inline fun
IntArray.sumOf(selector: (Int) -> Int): Int {\n var sum: Int = 0.toInt()\n for (element in this) {\n sum +=
selector(element)\n } \n return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfInt")\n@kotlin.internal.InlineOnly\npublic inline fun
LongArray.sumOf(selector: (Long) -> Int): Int {\n var sum: Int = 0.toInt()\n for (element in this) {\n sum +=
selector(element)\n } \n return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfInt")\n@kotlin.internal.InlineOnly\npublic inline fun
FloatArray.sumOf(selector: (Float) -> Int): Int {\n var sum: Int = 0.toInt()\n for (element in this) {\n sum +=
selector(element)\n } \n return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfInt")\n@kotlin.internal.InlineOnly\npublic inline fun
DoubleArray.sumOf(selector: (Double) -> Int): Int {\n var sum: Int = 0.toInt()\n for (element in this) {\n
sum += selector(element)\n } \n return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfInt")\n@kotlin.internal.InlineOnly\npublic inline fun
BooleanArray.sumOf(selector: (Boolean) -> Int): Int {\n var sum: Int = 0.toInt()\n for (element in this) {\n
sum += selector(element)\n } \n return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfInt")\n@kotlin.internal.InlineOnly\npublic inline fun
CharArray.sumOf(selector: (Char) -> Int): Int {\n var sum: Int = 0.toInt()\n for (element in this) {\n sum +=
selector(element)\n } \n return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfLong")\n@kotlin.internal.InlineOnly\npublic inline fun
<T> Array<out T>.sumOf(selector: (T) -> Long): Long {\n var sum: Long = 0.toLong()\n for (element in this)
{\n sum += selector(element)\n } \n return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfLong")\n@kotlin.internal.InlineOnly\npublic inline fun
ByteArray.sumOf(selector: (Byte) -> Long): Long {\n var sum: Long = 0.toLong()\n for (element in this) {\n
sum += selector(element)\n } \n return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution

```

```

ByLambdaReturnType\n@kotlin.jvm.JvmName("\sumOfLong")\n@kotlin.internal.InlineOnly\npublic inline fun
ShortArray.sumOf(selector: (Short) -> Long): Long {\n  var sum: Long = 0.toLong()\n  for (element in this) {\n
  sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\sumOfLong")\n@kotlin.internal.InlineOnly\npublic inline fun
IntArray.sumOf(selector: (Int) -> Long): Long {\n  var sum: Long = 0.toLong()\n  for (element in this) {\n
sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\sumOfLong")\n@kotlin.internal.InlineOnly\npublic inline fun
LongArray.sumOf(selector: (Long) -> Long): Long {\n  var sum: Long = 0.toLong()\n  for (element in this) {\n
sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\sumOfLong")\n@kotlin.internal.InlineOnly\npublic inline fun
FloatArray.sumOf(selector: (Float) -> Long): Long {\n  var sum: Long = 0.toLong()\n  for (element in this) {\n
sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\sumOfLong")\n@kotlin.internal.InlineOnly\npublic inline fun
DoubleArray.sumOf(selector: (Double) -> Long): Long {\n  var sum: Long = 0.toLong()\n  for (element in this)
{\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\sumOfLong")\n@kotlin.internal.InlineOnly\npublic inline fun
BooleanArray.sumOf(selector: (Boolean) -> Long): Long {\n  var sum: Long = 0.toLong()\n  for (element in this)
{\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the array.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\sumOfLong")\n@kotlin.internal.InlineOnly\npublic inline fun
CharArray.sumOf(selector: (Char) -> Long): Long {\n  var sum: Long = 0.toLong()\n  for (element in this) {\n
sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the array.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\sumOfUInt")\n@WasExperimental(ExperimentalUnsignedType
s::class)\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out T>.sumOf(selector: (T) -> UInt): UInt {\n
var sum: UInt = 0.toUInt()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return
sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the
array.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\sumOfUInt")\n@WasExperimental(ExperimentalUnsignedType
s::class)\n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.sumOf(selector: (Byte) -> UInt): UInt {\n  var
sum: UInt = 0.toUInt()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the array.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("\sumOfUInt")\n@WasExperimental(ExperimentalUnsignedType
s::class)\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.sumOf(selector: (Short) -> UInt): UInt {\n  var

```

```

sum: UInt = 0.toUInt()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the array.\n */\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfUInt")\n@WasExperimental(ExperimentalUnsignedType\ns::class)\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.sumOf(selector: (Int) -> UInt): UInt {\n  var\n  sum: UInt = 0.toUInt()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the array.\n */\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedType\ns::class)\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.sumOf(selector: (Long) -> UInt): UInt {\n  var\n  sum: UInt = 0.toUInt()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the array.\n */\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedType\ns::class)\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.sumOf(selector: (Float) -> UInt): UInt {\n  var\n  sum: UInt = 0.toUInt()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the array.\n */\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedType\ns::class)\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.sumOf(selector: (Double) -> UInt): UInt {\n  var\n  var sum: UInt = 0.toUInt()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return\n  sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the\n  array.\n */\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfUInt")\n@WasExperimental(ExperimentalUnsignedType\ns::class)\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.sumOf(selector: (Boolean) -> UInt): UInt\n{\n  var sum: UInt = 0.toUInt()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return\n  sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the\n  array.\n */\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedType\ns::class)\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.sumOf(selector: (Char) -> UInt): UInt {\n  var\n  sum: UInt = 0.toUInt()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the array.\n */\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedType\ns::class)\n@kotlin.internal.InlineOnly\npublic inline fun <T> Array<out T>.sumOf(selector: (T) -> ULong): ULong\n{\n  var sum: ULong = 0.toULong()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in\n  the array.\n */\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedType\ns::class)\n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.sumOf(selector: (Byte) -> ULong): ULong\n{\n  var sum: ULong = 0.toULong()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return\n  sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the\n  array.\n */\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedType\n
```

```

pes::class)\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.sumOf(selector: (Short) -> ULong): ULong
{\n  var sum: ULong = 0.toULong()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return
sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the
array.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedTy
pes::class)\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.sumOf(selector: (Int) -> ULong): ULong {\n
var sum: ULong = 0.toULong()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return
sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the
array.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedTy
pes::class)\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.sumOf(selector: (Long) -> ULong): ULong
{\n  var sum: ULong = 0.toULong()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return
sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the
array.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedTy
pes::class)\n@kotlin.internal.InlineOnly\npublic inline fun FloatArray.sumOf(selector: (Float) -> ULong): ULong
{\n  var sum: ULong = 0.toULong()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return
sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the
array.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedTy
pes::class)\n@kotlin.internal.InlineOnly\npublic inline fun DoubleArray.sumOf(selector: (Double) -> ULong):
ULong {\n  var sum: ULong = 0.toULong()\n  for (element in this) {\n    sum += selector(element)\n  }\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in
the array.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedTy
pes::class)\n@kotlin.internal.InlineOnly\npublic inline fun BooleanArray.sumOf(selector: (Boolean) -> ULong):
ULong {\n  var sum: ULong = 0.toULong()\n  for (element in this) {\n    sum += selector(element)\n  }\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in
the array.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedTy
pes::class)\n@kotlin.internal.InlineOnly\npublic inline fun CharArray.sumOf(selector: (Char) -> ULong): ULong
{\n  var sum: ULong = 0.toULong()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return
sum\n}\n\n/**\n * Returns an original collection containing all the non-`null` elements, throwing an
[IllegalArgumentException] if there are any `null` elements.\n
*\npublic fun <T : Any>
Array<T?>.requireNonNulls(): Array<T> {\n  for (element in this) {\n    if (element == null) {\n      throw
IllegalArgumentException("null element found in $this.")\n    }\n  }\n
@Suppress("UNCHECKED_CAST")\n  return this as Array<T>\n}\n\n/**\n * Splits the original array into pair
of lists,\n * where *first* list contains elements for which [predicate] yielded `true`,\n * while *second* list contains
elements for which [predicate] yielded `false`.\n
*\n * @sample
samples.collections.Arrays.Transformations.partitionArrayOfPrimitives\n
*\npublic inline fun <T> Array<out
T>.partition(predicate: (T) -> Boolean): Pair<List<T>, List<T>> {\n  val first = ArrayList<T>()\n  val second =
ArrayList<T>()\n  for (element in this) {\n    if (predicate(element)) {\n      first.add(element)\n    } else {\n

```

```

second.add(element)\n    }\n    }\n    return Pair(first, second)\n}\n\n/**\n * Splits the original array into pair
of lists,\n * where *first* list contains elements for which [predicate] yielded `true`,\n * while *second* list contains
elements for which [predicate] yielded `false`.\n * \n * @sample
samples.collections.Arrays.Transformations.partitionArrayOfPrimitives\n */\npublic inline fun
ByteArray.partition(predicate: (Byte) -> Boolean): Pair<List<Byte>, List<Byte>> {\n    val first =
ArrayList<Byte>()\n    val second = ArrayList<Byte>()\n    for (element in this) {\n        if (predicate(element)) {\n
        first.add(element)\n        } else {\n            second.add(element)\n        }\n    }\n    return Pair(first,
second)\n}\n\n/**\n * Splits the original array into pair of lists,\n * where *first* list contains elements for which
[predicate] yielded `true`,\n * while *second* list contains elements for which [predicate] yielded `false`.\n * \n *
@sample samples.collections.Arrays.Transformations.partitionArrayOfPrimitives\n */\npublic inline fun
ShortArray.partition(predicate: (Short) -> Boolean): Pair<List<Short>, List<Short>> {\n    val first =
ArrayList<Short>()\n    val second = ArrayList<Short>()\n    for (element in this) {\n        if (predicate(element)) {\n
        first.add(element)\n        } else {\n            second.add(element)\n        }\n    }\n    return Pair(first,
second)\n}\n\n/**\n * Splits the original array into pair of lists,\n * where *first* list contains elements for which
[predicate] yielded `true`,\n * while *second* list contains elements for which [predicate] yielded `false`.\n * \n *
@sample samples.collections.Arrays.Transformations.partitionArrayOfPrimitives\n */\npublic inline fun
IntArray.partition(predicate: (Int) -> Boolean): Pair<List<Int>, List<Int>> {\n    val first = ArrayList<Int>()\n    val
second = ArrayList<Int>()\n    for (element in this) {\n        if (predicate(element)) {\n            first.add(element)\n
        } else {\n            second.add(element)\n        }\n    }\n    return Pair(first, second)\n}\n\n/**\n * Splits the original
array into pair of lists,\n * where *first* list contains elements for which [predicate] yielded `true`,\n * while
*second* list contains elements for which [predicate] yielded `false`.\n * \n * @sample
samples.collections.Arrays.Transformations.partitionArrayOfPrimitives\n */\npublic inline fun
LongArray.partition(predicate: (Long) -> Boolean): Pair<List<Long>, List<Long>> {\n    val first =
ArrayList<Long>()\n    val second = ArrayList<Long>()\n    for (element in this) {\n        if (predicate(element)) {\n
        first.add(element)\n        } else {\n            second.add(element)\n        }\n    }\n    return Pair(first,
second)\n}\n\n/**\n * Splits the original array into pair of lists,\n * where *first* list contains elements for which
[predicate] yielded `true`,\n * while *second* list contains elements for which [predicate] yielded `false`.\n * \n *
@sample samples.collections.Arrays.Transformations.partitionArrayOfPrimitives\n */\npublic inline fun
FloatArray.partition(predicate: (Float) -> Boolean): Pair<List<Float>, List<Float>> {\n    val first =
ArrayList<Float>()\n    val second = ArrayList<Float>()\n    for (element in this) {\n        if (predicate(element)) {\n
        first.add(element)\n        } else {\n            second.add(element)\n        }\n    }\n    return Pair(first,
second)\n}\n\n/**\n * Splits the original array into pair of lists,\n * where *first* list contains elements for which
[predicate] yielded `true`,\n * while *second* list contains elements for which [predicate] yielded `false`.\n * \n *
@sample samples.collections.Arrays.Transformations.partitionArrayOfPrimitives\n */\npublic inline fun
DoubleArray.partition(predicate: (Double) -> Boolean): Pair<List<Double>, List<Double>> {\n    val first =
ArrayList<Double>()\n    val second = ArrayList<Double>()\n    for (element in this) {\n        if (predicate(element))
{\n            first.add(element)\n        } else {\n            second.add(element)\n        }\n    }\n    return Pair(first,
second)\n}\n\n/**\n * Splits the original array into pair of lists,\n * where *first* list contains elements for which
[predicate] yielded `true`,\n * while *second* list contains elements for which [predicate] yielded `false`.\n * \n *
@sample samples.collections.Arrays.Transformations.partitionArrayOfPrimitives\n */\npublic inline fun
BooleanArray.partition(predicate: (Boolean) -> Boolean): Pair<List<Boolean>, List<Boolean>> {\n    val first =
ArrayList<Boolean>()\n    val second = ArrayList<Boolean>()\n    for (element in this) {\n        if
(predicate(element)) {\n            first.add(element)\n        } else {\n            second.add(element)\n        }\n    }\n
return Pair(first, second)\n}\n\n/**\n * Splits the original array into pair of lists,\n * where *first* list contains
elements for which [predicate] yielded `true`,\n * while *second* list contains elements for which [predicate]
yielded `false`.\n * \n * @sample samples.collections.Arrays.Transformations.partitionArrayOfPrimitives\n
*/\npublic inline fun CharArray.partition(predicate: (Char) -> Boolean): Pair<List<Char>, List<Char>> {\n    val
first = ArrayList<Char>()\n    val second = ArrayList<Char>()\n    for (element in this) {\n        if

```

```

(predicate(element)) {\n      first.add(element)\n    } else {\n      second.add(element)\n    }\n  }\n  return Pair(first, second)\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` array and the [other]\n array with the same index.\n * The returned list has length of the shortest collection.\n * \n * @sample\n samples.collections.Iterables.Operations.zipIterable\n */\npublic infix fun <T, R> Array<out T>.zip(other:\n Array<out R>): List<Pair<T, R>> {\n  return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs\n built from the elements of `this` array and the [other] array with the same index.\n * The returned list has length of\n the shortest collection.\n * \n * @sample\n samples.collections.Iterables.Operations.zipIterable\n */\npublic infix fun\n <R> ByteArray.zip(other: Array<out R>): List<Pair<Byte, R>> {\n  return zip(other) { t1, t2 -> t1 to t2\n }\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` array and the [other] array with the same\n index.\n * The returned list has length of the shortest collection.\n * \n * @sample\n samples.collections.Iterables.Operations.zipIterable\n */\npublic infix fun <R> ShortArray.zip(other: Array<out\n R>): List<Pair<Short, R>> {\n  return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built from\n the elements of `this` array and the [other] array with the same index.\n * The returned list has length of the shortest\n collection.\n * \n * @sample\n samples.collections.Iterables.Operations.zipIterable\n */\npublic infix fun <R>\n IntArray.zip(other: Array<out R>): List<Pair<Int, R>> {\n  return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` array and the [other] array with the same index.\n * The\n returned list has length of the shortest collection.\n * \n * @sample\n samples.collections.Iterables.Operations.zipIterable\n */\npublic infix fun <R> LongArray.zip(other: Array<out\n R>): List<Pair<Long, R>> {\n  return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built from\n the elements of `this` array and the [other] array with the same index.\n * The returned list has length of the shortest\n collection.\n * \n * @sample\n samples.collections.Iterables.Operations.zipIterable\n */\npublic infix fun <R>\n FloatArray.zip(other: Array<out R>): List<Pair<Float, R>> {\n  return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` array and the [other] array with the same index.\n * The\n returned list has length of the shortest collection.\n * \n * @sample\n samples.collections.Iterables.Operations.zipIterable\n */\npublic infix fun <R> DoubleArray.zip(other: Array<out\n R>): List<Pair<Double, R>> {\n  return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built\n from the elements of `this` array and the [other] array with the same index.\n * The returned list has length of the\n shortest collection.\n * \n * @sample\n samples.collections.Iterables.Operations.zipIterable\n */\npublic infix fun <R>\n BooleanArray.zip(other: Array<out R>): List<Pair<Boolean, R>> {\n  return zip(other) { t1, t2 -> t1 to t2\n }\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` array and the [other] array with the same\n index.\n * The returned list has length of the shortest collection.\n * \n * @sample\n samples.collections.Iterables.Operations.zipIterable\n */\npublic infix fun <R> CharArray.zip(other: Array<out R>):\n List<Pair<Char, R>> {\n  return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of values built from the\n elements of `this` array and the [other] array with the same index\n * using the provided [transform] function\n applied to each pair of elements.\n * The returned list has length of the shortest collection.\n * \n * @sample\n samples.collections.Iterables.Operations.zipIterableWithTransform\n */\npublic inline fun <T, R, V> Array<out\n T>.zip(other: Array<out R>, transform: (a: T, b: R) -> V): List<V> {\n  val size = minOf(size, other.size)\n  val\n list = ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n  }\n  return\n list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array with the same\n index\n * using the provided [transform] function applied to each pair of elements.\n * The returned list has length\n of the shortest collection.\n * \n * @sample\n samples.collections.Iterables.Operations.zipIterableWithTransform\n */\npublic inline fun <R, V> ByteArray.zip(other: Array<out R>, transform: (a: Byte, b: R) -> V): List<V> {\n  val\n size = minOf(size, other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n  }\n  return list\n}\n\n/**\n * Returns a list of values built from the elements\n of `this` array and the [other] array with the same index\n * using the provided [transform] function applied to each\n pair of elements.\n * The returned list has length of the shortest collection.\n * \n * @sample\n samples.collections.Iterables.Operations.zipIterableWithTransform\n */\npublic inline fun <R, V>\n ShortArray.zip(other: Array<out R>, transform: (a: Short, b: R) -> V): List<V> {\n  val size = minOf(size,

```

```

other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n  }\n  return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array\n * with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned\n * list has length of the\n * shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n *\npublic inline fun <R, V>
IntArray.zip(other: Array<out R>, transform: (a: Int, b: R) -> V): List<V> {\n  val size = minOf(size, other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n  }\n  return
list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array with the same
index\n * using the provided [transform] function applied to each pair of elements.\n * The returned list has length
of the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\npublic inline fun <R, V> LongArray.zip(other: Array<out R>, transform: (a: Long, b: R) -> V): List<V> {\n
val size = minOf(size, other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n
list.add(transform(this[i], other[i]))\n  }\n  return list\n}\n\n/**\n * Returns a list of values built from the elements
of `this` array and the [other] array with the same index\n * using the provided [transform] function applied to each
pair of elements.\n * The returned list has length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n *\npublic inline fun <R, V>
FloatArray.zip(other: Array<out R>, transform: (a: Float, b: R) -> V): List<V> {\n  val size = minOf(size,
other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n
}\n  return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array
with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned
list has length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n *\npublic inline fun <R, V>
DoubleArray.zip(other: Array<out R>, transform: (a: Double, b: R) -> V): List<V> {\n  val size = minOf(size,
other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n
}\n  return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array
with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned
list has length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n *\npublic inline fun <R, V>
BooleanArray.zip(other: Array<out R>, transform: (a: Boolean, b: R) -> V): List<V> {\n  val size = minOf(size,
other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n
}\n  return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array
with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned
list has length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n *\npublic inline fun <R, V>
CharArray.zip(other: Array<out R>, transform: (a: Char, b: R) -> V): List<V> {\n  val size = minOf(size,
other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n
}\n  return list\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` collection and [other] array with
the same index.\n * The returned list has length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterable\n *\npublic infix fun <T, R> Array<out T>.zip(other:
Iterable<R>): List<Pair<T, R>> {\n  return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built
from the elements of `this` collection and [other] array with the same index.\n * The returned list has length of the
shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n *\npublic infix fun <R>
ByteArray.zip(other: Iterable<R>): List<Pair<Byte, R>> {\n  return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n *
Returns a list of pairs built from the elements of `this` collection and [other] array with the same index.\n * The
returned list has length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterable\n *\npublic infix fun <R> ShortArray.zip(other: Iterable<R>):
List<Pair<Short, R>> {\n  return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built from the
elements of `this` collection and [other] array with the same index.\n * The returned list has length of the shortest

```

```

collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n * \npublic infix fun <R>
IntArray.zip(other: Iterable<R>): List<Pair<Int, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n *
Returns a list of pairs built from the elements of `this` collection and [other] array with the same index.\n * The
returned list has length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterable\n * \npublic infix fun <R> LongArray.zip(other: Iterable<R>):
List<Pair<Long, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built from the
elements of `this` collection and [other] array with the same index.\n * The returned list has length of the shortest
collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n * \npublic infix fun <R>
FloatArray.zip(other: Iterable<R>): List<Pair<Float, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n *
Returns a list of pairs built from the elements of `this` collection and [other] array with the same index.\n * The
returned list has length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterable\n * \npublic infix fun <R> DoubleArray.zip(other:
Iterable<R>): List<Pair<Double, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs
built from the elements of `this` collection and [other] array with the same index.\n * The returned list has length of
the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n * \npublic infix fun
<R> BooleanArray.zip(other: Iterable<R>): List<Pair<Boolean, R>> {\n    return zip(other) { t1, t2 -> t1 to t2
}\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` collection and [other] array with the same
index.\n * The returned list has length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterable\n * \npublic infix fun <R> CharArray.zip(other: Iterable<R>):
List<Pair<Char, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of values built from the
elements of `this` array and the [other] collection with the same index\n * using the provided [transform] function
applied to each pair of elements.\n * The returned list has length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n * \npublic inline fun <T, R, V> Array<out
T>.zip(other: Iterable<R>, transform: (a: T, b: R) -> V): List<V> {\n    val arraySize = size\n    val list =
ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))\n    var i = 0\n    for (element in other) {\n
if (i >= arraySize) break\n        list.add(transform(this[i++], element))\n    }\n    return list\n}\n\n/**\n * Returns a
list of values built from the elements of `this` array and the [other] collection with the same index\n * using the
provided [transform] function applied to each pair of elements.\n * The returned list has length of the shortest
collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n * \npublic inline
fun <R, V> ByteArray.zip(other: Iterable<R>, transform: (a: Byte, b: R) -> V): List<V> {\n    val arraySize = size\n
val list = ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))\n    var i = 0\n    for (element in
other) {\n        if (i >= arraySize) break\n        list.add(transform(this[i++], element))\n    }\n    return list\n}\n\n/**\n *
Returns a list of values built from the elements of `this` array and the [other] collection with the same index\n *
using the provided [transform] function applied to each pair of elements.\n * The returned list has length of the
shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n * \npublic
inline fun <R, V> ShortArray.zip(other: Iterable<R>, transform: (a: Short, b: R) -> V): List<V> {\n    val arraySize =
size\n    val list = ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))\n    var i = 0\n    for
(element in other) {\n        if (i >= arraySize) break\n        list.add(transform(this[i++], element))\n    }\n    return
list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] collection with the
same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned list has
length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n * \npublic inline fun <R, V>
IntArray.zip(other: Iterable<R>, transform: (a: Int, b: R) -> V): List<V> {\n    val arraySize = size\n    val list =
ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))\n    var i = 0\n    for (element in other) {\n
if (i >= arraySize) break\n        list.add(transform(this[i++], element))\n    }\n    return list\n}\n\n/**\n * Returns a
list of values built from the elements of `this` array and the [other] collection with the same index\n * using the
provided [transform] function applied to each pair of elements.\n * The returned list has length of the shortest
collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n * \npublic inline

```

```

fun <R, V> LongArray.zip(other: Iterable<R>, transform: (a: Long, b: R) -> V): List<V> {
    val arraySize = size
    val list = ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))
    var i = 0
    for (element in other) {
        if (i >= arraySize) break
        list.add(transform(this[i++], element))
    }
    return list
}

/**
 * Returns a list of values built from the elements of `this` array and the [other] collection with the
 * same index
 * using the provided [transform] function applied to each pair of elements.
 * The returned list has
 * length of the shortest collection.
 * @sample
 */
samples.collections.Iterables.Operations.zipIterableWithTransform

/**
 * public inline fun <R, V>
 */
FloatArray.zip(other: Iterable<R>, transform: (a: Float, b: R) -> V): List<V> {
    val arraySize = size
    val list = ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))
    var i = 0
    for (element in other) {
        if (i >= arraySize) break
        list.add(transform(this[i++], element))
    }
    return list
}

/**
 * Returns a list of values built from the elements of `this` array and the [other] collection with the same index
 * using the
 * provided [transform] function applied to each pair of elements.
 * The returned list has length of the shortest
 * collection.
 * @sample
 */
samples.collections.Iterables.Operations.zipIterableWithTransform

/**
 * public inline
 */
fun <R, V> DoubleArray.zip(other: Iterable<R>, transform: (a: Double, b: R) -> V): List<V> {
    val arraySize = size
    val list = ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))
    var i = 0
    for (element in other) {
        if (i >= arraySize) break
        list.add(transform(this[i++], element))
    }
    return list
}

/**
 * Returns a list of values built from the elements of `this` array and the [other] collection with the
 * same index
 * using the provided [transform] function applied to each pair of elements.
 * The returned list has
 * length of the shortest collection.
 * @sample
 */
samples.collections.Iterables.Operations.zipIterableWithTransform

/**
 * public inline fun <R, V>
 */
BooleanArray.zip(other: Iterable<R>, transform: (a: Boolean, b: R) -> V): List<V> {
    val arraySize = size
    val list = ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))
    var i = 0
    for (element in other) {
        if (i >= arraySize) break
        list.add(transform(this[i++], element))
    }
    return list
}

/**
 * Returns a list of values built from the elements of `this` array and the [other] collection with the same index
 * using the
 * provided [transform] function applied to each pair of elements.
 * The returned list has length of the
 * shortest collection.
 * @sample
 */
samples.collections.Iterables.Operations.zipIterableWithTransform

/**
 * public inline fun <R, V>
 */
CharArray.zip(other: Iterable<R>, transform: (a: Char, b: R) -> V): List<V> {
    val arraySize = size
    val list = ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))
    var i = 0
    for (element in other) {
        if (i >= arraySize) break
        list.add(transform(this[i++], element))
    }
    return list
}

/**
 * Returns a list of pairs built from the elements of `this` array and the [other] array with the same
 * index.
 * The returned list has length of the shortest collection.
 * @sample
 */
samples.collections.Iterables.Operations.zipIterable

/**
 * public infix fun
 */
ByteArray.zip(other: ByteArray): List<Pair<Byte, Byte>> {
    return zip(other) { t1, t2 -> t1 to t2 }
}

/**
 * Returns a list of pairs built from
 * the elements of `this` array and the [other] array with the same index.
 * The returned list has length of the shortest
 * collection.
 * @sample
 */
samples.collections.Iterables.Operations.zipIterable

/**
 * public infix fun
 */
ShortArray.zip(other: ShortArray): List<Pair<Short, Short>> {
    return zip(other) { t1, t2 -> t1 to t2 }
}

/**
 * Returns a list of pairs built from the elements of `this` array and the [other] array with the same index.
 * The
 * returned list has length of the shortest collection.
 * @sample
 */
samples.collections.Iterables.Operations.zipIterable

/**
 * public infix fun
 */
IntArray.zip(other: IntArray): List<Pair<Int, Int>> {
    return zip(other) { t1, t2 -> t1 to t2 }
}

/**
 * Returns a list of pairs built from the
 * elements of `this` array and the [other] array with the same index.
 * The returned list has length of the shortest
 * collection.
 * @sample
 */
samples.collections.Iterables.Operations.zipIterable

/**
 * public infix fun
 */
LongArray.zip(other: LongArray): List<Pair<Long, Long>> {
    return zip(other) { t1, t2 -> t1 to t2 }
}

/**
 * Returns a list of pairs built from the elements of `this` array and the [other] array with the same index.
 * The
 * returned list has length of the shortest collection.
 * @sample
 */
samples.collections.Iterables.Operations.zipIterable

/**
 * public infix fun
 */
FloatArray.zip(other: FloatArray): List<Pair<Float, Float>> {
    return zip(other) { t1, t2 -> t1 to t2 }
}

/**
 * Returns a list of pairs built from
 * the elements of `this` array and the [other] array with the same index.
 * The returned list has length of the shortest
 */

```

```

collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n *^\npublic infix fun
DoubleArray.zip(other: DoubleArray): List<Pair<Double, Double>> {\n  return zip(other) { t1, t2 -> t1 to t2
}\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` array and the [other] array with the same
index.\n * The returned list has length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterable\n *^\npublic infix fun BooleanArray.zip(other: BooleanArray):
List<Pair<Boolean, Boolean>> {\n  return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built
from the elements of `this` array and the [other] array with the same index.\n * The returned list has length of the
shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n *^\npublic infix fun
CharArray.zip(other: CharArray): List<Pair<Char, Char>> {\n  return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n *
Returns a list of values built from the elements of `this` array and the [other] array with the same index\n * using the
provided [transform] function applied to each pair of elements.\n * The returned list has length of the shortest
array.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n *^\npublic inline fun
<V> ByteArray.zip(other: ByteArray, transform: (a: Byte, b: Byte) -> V): List<V> {\n  val size = minOf(size,
other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n
}\n  return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array
with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned
list has length of the shortest array.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n *^\npublic inline fun <V>
ShortArray.zip(other: ShortArray, transform: (a: Short, b: Short) -> V): List<V> {\n  val size = minOf(size,
other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n
}\n  return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array
with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned
list has length of the shortest array.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n *^\npublic inline fun <V> IntArray.zip(other:
IntArray, transform: (a: Int, b: Int) -> V): List<V> {\n  val size = minOf(size, other.size)\n  val list =
ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n  }\n  return
list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array with the same
index\n * using the provided [transform] function applied to each pair of elements.\n * The returned list has length
of the shortest array.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n
*^\npublic inline fun <V> LongArray.zip(other: LongArray, transform: (a: Long, b: Long) -> V): List<V> {\n  val
size = minOf(size, other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n
list.add(transform(this[i], other[i]))\n  }\n  return list\n}\n\n/**\n * Returns a list of values built from the elements
of `this` array and the [other] array with the same index\n * using the provided [transform] function applied to each
pair of elements.\n * The returned list has length of the shortest array.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n *^\npublic inline fun <V>
FloatArray.zip(other: FloatArray, transform: (a: Float, b: Float) -> V): List<V> {\n  val size = minOf(size,
other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n
}\n  return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array
with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned
list has length of the shortest array.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n *^\npublic inline fun <V>
DoubleArray.zip(other: DoubleArray, transform: (a: Double, b: Double) -> V): List<V> {\n  val size = minOf(size,
other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n    list.add(transform(this[i], other[i]))\n
}\n  return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array
with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned
list has length of the shortest array.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n *^\npublic inline fun <V>
BooleanArray.zip(other: BooleanArray, transform: (a: Boolean, b: Boolean) -> V): List<V> {\n  val size =

```

```

minOf(size, other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n      list.add(transform(this[i],
other[i]))\n  }\n  return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the
[other] array with the same index\n * using the provided [transform] function applied to each pair of elements.\n *
The returned list has length of the shortest array.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n */\npublic inline fun <V>
CharArray.zip(other: CharArray, transform: (a: Char, b: Char) -> V): List<V> {\n  val size = minOf(size,
other.size)\n  val list = ArrayList<V>(size)\n  for (i in 0 until size) {\n      list.add(transform(this[i], other[i]))\n
  }\n  return list\n}\n\n/**\n * Appends the string from all the elements separated using [separator] and using the
given [prefix] and [postfix] if supplied.\n * \n * If the collection could be huge, you can specify a non-negative value
of [limit], in which case only the first [limit]\n * elements will be appended, followed by the [truncated] string
(which defaults to "...").\n * \n * @sample samples.collections.Collections.Transformations.joinTo\n */\npublic fun
<T, A : Appendable> Array<out T>.joinTo(buffer: A, separator: CharSequence = "\", \"", prefix: CharSequence = "\",
postfix: CharSequence = "\", limit: Int = -1, truncated: CharSequence = "...\", transform: ((T) -> CharSequence)? =
null): A {\n  buffer.append(prefix)\n  var count = 0\n  for (element in this) {\n      if (++count > 1)
buffer.append(separator)\n      if (limit < 0 || count <= limit) {\n          buffer.appendElement(element, transform)\n
      } else break\n  }\n  if (limit >= 0 && count > limit) buffer.append(truncated)\n  buffer.append(postfix)\n
return buffer\n}\n\n/**\n * Appends the string from all the elements separated using [separator] and using the given
[prefix] and [postfix] if supplied.\n * \n * If the collection could be huge, you can specify a non-negative value of
[limit], in which case only the first [limit]\n * elements will be appended, followed by the [truncated] string (which
defaults to "...").\n * \n * @sample samples.collections.Collections.Transformations.joinTo\n */\npublic fun <A :
Appendable> ByteArray.joinTo(buffer: A, separator: CharSequence = "\", \"", prefix: CharSequence = "\", postfix:
CharSequence = "\", limit: Int = -1, truncated: CharSequence = "...\", transform: ((Byte) -> CharSequence)? =
null): A {\n  buffer.append(prefix)\n  var count = 0\n  for (element in this) {\n      if (++count > 1)
buffer.append(separator)\n      if (limit < 0 || count <= limit) {\n          if (transform != null)\nbuffer.append(transform(element))\n          else\nbuffer.append(element.toString())\n      } else break\n
  }\n  if (limit >= 0 && count > limit) buffer.append(truncated)\n  buffer.append(postfix)\n  return
buffer\n}\n\n/**\n * Appends the string from all the elements separated using [separator] and using the given
[prefix] and [postfix] if supplied.\n * \n * If the collection could be huge, you can specify a non-negative value of
[limit], in which case only the first [limit]\n * elements will be appended, followed by the [truncated] string (which
defaults to "...").\n * \n * @sample samples.collections.Collections.Transformations.joinTo\n */\npublic fun <A :
Appendable> ShortArray.joinTo(buffer: A, separator: CharSequence = "\", \"", prefix: CharSequence = "\", postfix:
CharSequence = "\", limit: Int = -1, truncated: CharSequence = "...\", transform: ((Short) -> CharSequence)? =
null): A {\n  buffer.append(prefix)\n  var count = 0\n  for (element in this) {\n      if (++count > 1)
buffer.append(separator)\n      if (limit < 0 || count <= limit) {\n          if (transform != null)\nbuffer.append(transform(element))\n          else\nbuffer.append(element.toString())\n      } else break\n
  }\n  if (limit >= 0 && count > limit) buffer.append(truncated)\n  buffer.append(postfix)\n  return
buffer\n}\n\n/**\n * Appends the string from all the elements separated using [separator] and using the given
[prefix] and [postfix] if supplied.\n * \n * If the collection could be huge, you can specify a non-negative value of
[limit], in which case only the first [limit]\n * elements will be appended, followed by the [truncated] string (which
defaults to "...").\n * \n * @sample samples.collections.Collections.Transformations.joinTo\n */\npublic fun <A :
Appendable> IntArray.joinTo(buffer: A, separator: CharSequence = "\", \"", prefix: CharSequence = "\", postfix:
CharSequence = "\", limit: Int = -1, truncated: CharSequence = "...\", transform: ((Int) -> CharSequence)? =
null): A {\n  buffer.append(prefix)\n  var count = 0\n  for (element in this) {\n      if (++count > 1)
buffer.append(separator)\n      if (limit < 0 || count <= limit) {\n          if (transform != null)\nbuffer.append(transform(element))\n          else\nbuffer.append(element.toString())\n      } else break\n
  }\n  if (limit >= 0 && count > limit) buffer.append(truncated)\n  buffer.append(postfix)\n  return
buffer\n}\n\n/**\n * Appends the string from all the elements separated using [separator] and using the given
[prefix] and [postfix] if supplied.\n * \n * If the collection could be huge, you can specify a non-negative value of

```

```

[limit], in which case only the first [limit]\n * elements will be appended, followed by the [truncated] string (which
defaults to "...").\n * \n * @sample samples.collections.Collections.Transformations.joinTo\n * \npublic fun <A :
Appendable> LongArray.joinTo(buffer: A, separator: CharSequence = "\", \", prefix: CharSequence = "\"", postfix:
CharSequence = "\"", limit: Int = -1, truncated: CharSequence = "...", transform: ((Long) -> CharSequence)? =
null): A {\n  buffer.append(prefix)\n  var count = 0\n  for (element in this) {\n    if (++count > 1)
buffer.append(separator)\n    if (limit < 0 || count <= limit) {\n      if (transform != null)\nbuffer.append(transform(element))\n      else\n        buffer.append(element.toString())\n    } else break\n  }\n  if (limit >= 0 && count > limit) buffer.append(truncated)\n  buffer.append(postfix)\n  return
buffer\n}\n\n/**\n * Appends the string from all the elements separated using [separator] and using the given
[prefix] and [postfix] if supplied.\n * \n * If the collection could be huge, you can specify a non-negative value of
[limit], in which case only the first [limit]\n * elements will be appended, followed by the [truncated] string (which
defaults to "...").\n * \n * @sample samples.collections.Collections.Transformations.joinTo\n * \npublic fun <A :
Appendable> FloatArray.joinTo(buffer: A, separator: CharSequence = "\", \", prefix: CharSequence = "\"", postfix:
CharSequence = "\"", limit: Int = -1, truncated: CharSequence = "...", transform: ((Float) -> CharSequence)? =
null): A {\n  buffer.append(prefix)\n  var count = 0\n  for (element in this) {\n    if (++count > 1)
buffer.append(separator)\n    if (limit < 0 || count <= limit) {\n      if (transform != null)\nbuffer.append(transform(element))\n      else\n        buffer.append(element.toString())\n    } else break\n  }\n  if (limit >= 0 && count > limit) buffer.append(truncated)\n  buffer.append(postfix)\n  return
buffer\n}\n\n/**\n * Appends the string from all the elements separated using [separator] and using the given
[prefix] and [postfix] if supplied.\n * \n * If the collection could be huge, you can specify a non-negative value of
[limit], in which case only the first [limit]\n * elements will be appended, followed by the [truncated] string (which
defaults to "...").\n * \n * @sample samples.collections.Collections.Transformations.joinTo\n * \npublic fun <A :
Appendable> DoubleArray.joinTo(buffer: A, separator: CharSequence = "\", \", prefix: CharSequence = "\"", postfix:
CharSequence = "\"", limit: Int = -1, truncated: CharSequence = "...", transform: ((Double) -> CharSequence)? =
null): A {\n  buffer.append(prefix)\n  var count = 0\n  for (element in this) {\n    if (++count > 1)
buffer.append(separator)\n    if (limit < 0 || count <= limit) {\n      if (transform != null)\nbuffer.append(transform(element))\n      else\n        buffer.append(element.toString())\n    } else break\n  }\n  if (limit >= 0 && count > limit) buffer.append(truncated)\n  buffer.append(postfix)\n  return
buffer\n}\n\n/**\n * Appends the string from all the elements separated using [separator] and using the given
[prefix] and [postfix] if supplied.\n * \n * If the collection could be huge, you can specify a non-negative value of
[limit], in which case only the first [limit]\n * elements will be appended, followed by the [truncated] string (which
defaults to "...").\n * \n * @sample samples.collections.Collections.Transformations.joinTo\n * \npublic fun <A :
Appendable> BooleanArray.joinTo(buffer: A, separator: CharSequence = "\", \", prefix: CharSequence = "\"",
postfix: CharSequence = "\"", limit: Int = -1, truncated: CharSequence = "...", transform: ((Boolean) ->
CharSequence)? = null): A {\n  buffer.append(prefix)\n  var count = 0\n  for (element in this) {\n    if (++count
> 1) buffer.append(separator)\n    if (limit < 0 || count <= limit) {\n      if (transform != null)\nbuffer.append(transform(element))\n      else\n        buffer.append(element.toString())\n    } else break\n  }\n  if (limit >= 0 && count > limit) buffer.append(truncated)\n  buffer.append(postfix)\n  return
buffer\n}\n\n/**\n * Appends the string from all the elements separated using [separator] and using the given
[prefix] and [postfix] if supplied.\n * \n * If the collection could be huge, you can specify a non-negative value of
[limit], in which case only the first [limit]\n * elements will be appended, followed by the [truncated] string (which
defaults to "...").\n * \n * @sample samples.collections.Collections.Transformations.joinTo\n * \npublic fun <A :
Appendable> CharArray.joinTo(buffer: A, separator: CharSequence = "\", \", prefix: CharSequence = "\"", postfix:
CharSequence = "\"", limit: Int = -1, truncated: CharSequence = "...", transform: ((Char) -> CharSequence)? =
null): A {\n  buffer.append(prefix)\n  var count = 0\n  for (element in this) {\n    if (++count > 1)
buffer.append(separator)\n    if (limit < 0 || count <= limit) {\n      if (transform != null)\nbuffer.append(transform(element))\n      else\n        buffer.append(element)\n    } else break\n  }\n  if
(limit >= 0 && count > limit) buffer.append(truncated)\n  buffer.append(postfix)\n  return buffer\n}\n\n/**\n *

```

Creates a string from all the elements separated using [separator] and using the given [prefix] and [postfix] if supplied.

`String.joinToString(separator: CharSequence = "\", \", prefix: CharSequence = \"\", postfix: CharSequence = \"\", limit: Int = -1, truncated: CharSequence = \"...\", transform: ((T) -> CharSequence)? = null): String`

`return joinTo(StringBuilder(), separator, prefix, postfix, limit, truncated, transform).toString()`

Creates a string from all the elements separated using [separator] and using the given [prefix] and [postfix] if supplied.

`String.joinToString(separator: CharSequence = "\", \", prefix: CharSequence = \"\", postfix: CharSequence = \"\", limit: Int = -1, truncated: CharSequence = \"...\", transform: ((Byte) -> CharSequence)? = null): String`

`return joinTo(StringBuilder(), separator, prefix, postfix, limit, truncated, transform).toString()`

Creates a string from all the elements separated using [separator] and using the given [prefix] and [postfix] if supplied.

`String.joinToString(separator: CharSequence = "\", \", prefix: CharSequence = \"\", postfix: CharSequence = \"\", limit: Int = -1, truncated: CharSequence = \"...\", transform: ((Short) -> CharSequence)? = null): String`

`return joinTo(StringBuilder(), separator, prefix, postfix, limit, truncated, transform).toString()`

Creates a string from all the elements separated using [separator] and using the given [prefix] and [postfix] if supplied.

`String.joinToString(separator: CharSequence = "\", \", prefix: CharSequence = \"\", postfix: CharSequence = \"\", limit: Int = -1, truncated: CharSequence = \"...\", transform: ((Int) -> CharSequence)? = null): String`

`return joinTo(StringBuilder(), separator, prefix, postfix, limit, truncated, transform).toString()`

Creates a string from all the elements separated using [separator] and using the given [prefix] and [postfix] if supplied.

`String.joinToString(separator: CharSequence = "\", \", prefix: CharSequence = \"\", postfix: CharSequence = \"\", limit: Int = -1, truncated: CharSequence = \"...\", transform: ((Long) -> CharSequence)? = null): String`

`return joinTo(StringBuilder(), separator, prefix, postfix, limit, truncated, transform).toString()`

Creates a string from all the elements separated using [separator] and using the given [prefix] and [postfix] if supplied.

`String.joinToString(separator: CharSequence = "\", \", prefix: CharSequence = \"\", postfix: CharSequence = \"\", limit: Int = -1, truncated: CharSequence = \"...\", transform: ((Float) -> CharSequence)? = null): String`

`return joinTo(StringBuilder(), separator, prefix, postfix, limit, truncated, transform).toString()`

Creates a string from all the elements separated using [separator] and using the given [prefix] and [postfix] if supplied.

`String.joinToString(separator: CharSequence = "\", \", prefix: CharSequence = \"\", postfix: CharSequence = \"\", limit: Int = -1, truncated: CharSequence = \"...\", transform: ((Double) -> CharSequence)? = null): String`

`return joinTo(StringBuilder(), separator, prefix, postfix, limit, truncated, transform).toString()`

```

separator, prefix, postfix, limit, truncated, transform).toString()\n\n/**\n * Creates a string from all the elements
separated using [separator] and using the given [prefix] and [postfix] if supplied.\n * \n * If the collection could be
huge, you can specify a non-negative value of [limit], in which case only the first [limit]\n * elements will be
appended, followed by the [truncated] string (which defaults to "...").\n * \n * @sample
samples.collections.Collections.Transformations.joinToString\n\npublic fun
BooleanArray.joinToString(separator: CharSequence = "\", \"", prefix: CharSequence = "\", postfix: CharSequence =
\"", limit: Int = -1, truncated: CharSequence = "...\", transform: ((Boolean) -> CharSequence)? = null): String {\n
return joinTo(StringBuilder(), separator, prefix, postfix, limit, truncated, transform).toString()\n\n}\n\n/**\n * Creates
a string from all the elements separated using [separator] and using the given [prefix] and [postfix] if supplied.\n * \n
* If the collection could be huge, you can specify a non-negative value of [limit], in which case only the first
[limit]\n * elements will be appended, followed by the [truncated] string (which defaults to "...").\n * \n * @sample
samples.collections.Collections.Transformations.joinToString\n\npublic fun CharArray.joinToString(separator:
CharSequence = "\", \"", prefix: CharSequence = "\", postfix: CharSequence = "\", limit: Int = -1, truncated:
CharSequence = "...\", transform: ((Char) -> CharSequence)? = null): String {\n
return joinTo(StringBuilder(), separator, prefix, postfix, limit, truncated, transform).toString()\n\n}\n\n/**\n * Creates an [Iterable] instance that
wraps the original array returning its elements when being iterated.\n\npublic fun <T> Array<out T>.asIterable():
Iterable<T> {\n
if (isEmpty()) return emptyList()\n
return Iterable { this.iterator() }\n\n}\n\n/**\n * Creates an
[Iterable] instance that wraps the original array returning its elements when being iterated.\n\npublic fun
ByteArray.asIterable(): Iterable<Byte> {\n
if (isEmpty()) return emptyList()\n
return Iterable { this.iterator() }\n\n}\n\n/**\n * Creates an [Iterable] instance that wraps the original array returning its elements when being
iterated.\n\npublic fun ShortArray.asIterable(): Iterable<Short> {\n
if (isEmpty()) return emptyList()\n
return Iterable { this.iterator() }\n\n}\n\n/**\n * Creates an [Iterable] instance that wraps the original array returning its
elements when being iterated.\n\npublic fun IntArray.asIterable(): Iterable<Int> {\n
if (isEmpty()) return emptyList()\n
return Iterable { this.iterator() }\n\n}\n\n/**\n * Creates an [Iterable] instance that wraps the original
array returning its elements when being iterated.\n\npublic fun LongArray.asIterable(): Iterable<Long> {\n
if (isEmpty()) return emptyList()\n
return Iterable { this.iterator() }\n\n}\n\n/**\n * Creates an [Iterable] instance that
wraps the original array returning its elements when being iterated.\n\npublic fun FloatArray.asIterable():
Iterable<Float> {\n
if (isEmpty()) return emptyList()\n
return Iterable { this.iterator() }\n\n}\n\n/**\n * Creates an
[Iterable] instance that wraps the original array returning its elements when being iterated.\n\npublic fun
DoubleArray.asIterable(): Iterable<Double> {\n
if (isEmpty()) return emptyList()\n
return Iterable { this.iterator() }\n\n}\n\n/**\n * Creates an [Iterable] instance that wraps the original array returning its elements when
being iterated.\n\npublic fun BooleanArray.asIterable(): Iterable<Boolean> {\n
if (isEmpty()) return emptyList()\n
return Iterable { this.iterator() }\n\n}\n\n/**\n * Creates an [Iterable] instance that wraps the original
array returning its elements when being iterated.\n\npublic fun CharArray.asIterable(): Iterable<Char> {\n
if (isEmpty()) return emptyList()\n
return Iterable { this.iterator() }\n\n}\n\n/**\n * Creates a [Sequence] instance that
wraps the original array returning its elements when being iterated.\n * \n * @sample
samples.collections.Sequences.Building.sequenceFromArray\n\npublic fun <T> Array<out T>.asSequence():
Sequence<T> {\n
if (isEmpty()) return emptySequence()\n
return Sequence { this.iterator() }\n\n}\n\n/**\n *
Creates a [Sequence] instance that wraps the original array returning its elements when being iterated.\n * \n *
@sample samples.collections.Sequences.Building.sequenceFromArray\n\npublic fun ByteArray.asSequence():
Sequence<Byte> {\n
if (isEmpty()) return emptySequence()\n
return Sequence { this.iterator() }\n\n}\n\n/**\n *
Creates a [Sequence] instance that wraps the original array returning its elements when being iterated.\n * \n *
@sample samples.collections.Sequences.Building.sequenceFromArray\n\npublic fun ShortArray.asSequence():
Sequence<Short> {\n
if (isEmpty()) return emptySequence()\n
return Sequence { this.iterator() }\n\n}\n\n/**\n *
Creates a [Sequence] instance that wraps the original array returning its elements when being iterated.\n * \n *
@sample samples.collections.Sequences.Building.sequenceFromArray\n\npublic fun IntArray.asSequence():
Sequence<Int> {\n
if (isEmpty()) return emptySequence()\n
return Sequence { this.iterator() }\n\n}\n\n/**\n *
Creates a [Sequence] instance that wraps the original array returning its elements when being iterated.\n * \n *

```

```

@sample samples.collections.Sequences.Building.sequenceFromArray\n *\npublic fun LongArray.asSequence():
Sequence<Long> {\n if (isEmpty()) return emptySequence()\n return Sequence { this.iterator() }\n}\n\n/**\n *
Creates a [Sequence] instance that wraps the original array returning its elements when being iterated.\n * \n *
@sample samples.collections.Sequences.Building.sequenceFromArray\n *\npublic fun FloatArray.asSequence():
Sequence<Float> {\n if (isEmpty()) return emptySequence()\n return Sequence { this.iterator() }\n}\n\n/**\n *
Creates a [Sequence] instance that wraps the original array returning its elements when being iterated.\n * \n *
@sample samples.collections.Sequences.Building.sequenceFromArray\n *\npublic fun DoubleArray.asSequence():
Sequence<Double> {\n if (isEmpty()) return emptySequence()\n return Sequence { this.iterator() }\n}\n\n/**\n *
Creates a [Sequence] instance that wraps the original array returning its elements when being iterated.\n * \n *
@sample samples.collections.Sequences.Building.sequenceFromArray\n *\npublic fun
BooleanArray.asSequence(): Sequence<Boolean> {\n if (isEmpty()) return emptySequence()\n return Sequence
{ this.iterator() }\n}\n\n/**\n * Creates a [Sequence] instance that wraps the original array returning its elements
when being iterated.\n * \n * @sample samples.collections.Sequences.Building.sequenceFromArray\n *\npublic
fun CharArray.asSequence(): Sequence<Char> {\n if (isEmpty()) return emptySequence()\n return Sequence {
this.iterator() }\n}\n\n/**\n * Returns an average value of elements in the array.\n
*\n*\n@kotlin.jvm.JvmName("averageOfByte")\npublic fun Array<out Byte>.average(): Double {\n var sum:
Double = 0.0\n var count: Int = 0\n for (element in this) {\n sum += element\n ++count\n }\n return
if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns an average value of elements in the array.\n
*\n*\n@kotlin.jvm.JvmName("averageOfShort")\npublic fun Array<out Short>.average(): Double {\n var sum:
Double = 0.0\n var count: Int = 0\n for (element in this) {\n sum += element\n ++count\n }\n return
if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns an average value of elements in the array.\n
*\n*\n@kotlin.jvm.JvmName("averageOfInt")\npublic fun Array<out Int>.average(): Double {\n var sum: Double
= 0.0\n var count: Int = 0\n for (element in this) {\n sum += element\n ++count\n }\n return if (count
== 0) Double.NaN else sum / count\n}\n\n/**\n * Returns an average value of elements in the array.\n
*\n*\n@kotlin.jvm.JvmName("averageOfLong")\npublic fun Array<out Long>.average(): Double {\n var sum:
Double = 0.0\n var count: Int = 0\n for (element in this) {\n sum += element\n ++count\n }\n return
if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns an average value of elements in the array.\n
*\n*\n@kotlin.jvm.JvmName("averageOfFloat")\npublic fun Array<out Float>.average(): Double {\n var sum:
Double = 0.0\n var count: Int = 0\n for (element in this) {\n sum += element\n ++count\n }\n return
if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns an average value of elements in the array.\n
*\n*\n@kotlin.jvm.JvmName("averageOfDouble")\npublic fun Array<out Double>.average(): Double {\n var sum:
Double = 0.0\n var count: Int = 0\n for (element in this) {\n sum += element\n ++count\n }\n return
if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns an average value of elements in the array.\n
*\n*\npublic fun ByteArray.average(): Double {\n var sum: Double = 0.0\n var count: Int = 0\n for (element in
this) {\n sum += element\n ++count\n }\n return if (count == 0) Double.NaN else sum /
count\n}\n\n/**\n * Returns an average value of elements in the array.\n *\n*\npublic fun ShortArray.average():
Double {\n var sum: Double = 0.0\n var count: Int = 0\n for (element in this) {\n sum += element\n
++count\n }\n return if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns an average value of
elements in the array.\n *\n*\npublic fun IntArray.average(): Double {\n var sum: Double = 0.0\n var count: Int =
0\n for (element in this) {\n sum += element\n ++count\n }\n return if (count == 0) Double.NaN else
sum / count\n}\n\n/**\n * Returns an average value of elements in the array.\n *\n*\npublic fun LongArray.average():
Double {\n var sum: Double = 0.0\n var count: Int = 0\n for (element in this) {\n sum += element\n
++count\n }\n return if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns an average value of
elements in the array.\n *\n*\npublic fun FloatArray.average(): Double {\n var sum: Double = 0.0\n var count: Int
= 0\n for (element in this) {\n sum += element\n ++count\n }\n return if (count == 0) Double.NaN
else sum / count\n}\n\n/**\n * Returns an average value of elements in the array.\n *\n*\npublic fun
DoubleArray.average(): Double {\n var sum: Double = 0.0\n var count: Int = 0\n for (element in this) {\n
sum += element\n ++count\n }\n return if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns

```

```

the sum of all elements in the array.\n *^@kotlin.jvm.JvmName("sumOfByte")\npublic fun Array<out
Byte>.sum(): Int {\n    var sum: Int = 0\n    for (element in this) {\n        sum += element\n    }\n    return
sum\n}\n\n/**\n * Returns the sum of all elements in the array.\n
*^@kotlin.jvm.JvmName("sumOfShort")\npublic fun Array<out Short>.sum(): Int {\n    var sum: Int = 0\n    for
(element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in the
array.\n *^@kotlin.jvm.JvmName("sumOfInt")\npublic fun Array<out Int>.sum(): Int {\n    var sum: Int = 0\n
for (element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in
the array.\n *^@kotlin.jvm.JvmName("sumOfLong")\npublic fun Array<out Long>.sum(): Long {\n    var sum:
Long = 0L\n    for (element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of
all elements in the array.\n *^@kotlin.jvm.JvmName("sumOfFloat")\npublic fun Array<out Float>.sum(): Float
{\n    var sum: Float = 0.0f\n    for (element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n
* Returns the sum of all elements in the array.\n *^@kotlin.jvm.JvmName("sumOfDouble")\npublic fun Array<out
Double>.sum(): Double {\n    var sum: Double = 0.0\n    for (element in this) {\n        sum += element\n    }\n
return sum\n}\n\n/**\n * Returns the sum of all elements in the array.\n *^public fun ByteArray.sum(): Int {\n
var sum: Int = 0\n    for (element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the
sum of all elements in the array.\n *^public fun ShortArray.sum(): Int {\n    var sum: Int = 0\n    for (element
in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in the array.\n
*^public fun IntArray.sum(): Int {\n    var sum: Int = 0\n    for (element in this) {\n        sum += element\n    }\n
return sum\n}\n\n/**\n * Returns the sum of all elements in the array.\n *^public fun LongArray.sum(): Long {\n
var sum: Long = 0L\n    for (element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the
sum of all elements in the array.\n *^public fun FloatArray.sum(): Float {\n    var sum: Float = 0.0f\n    for
(element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in the
array.\n *^public fun DoubleArray.sum(): Double {\n    var sum: Double = 0.0\n    for (element in this) {\n
sum += element\n    }\n    return sum\n}\n\n"/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n
*^@n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("RangesKt")\n\npackage
kotlin.ranges\n\n/\n// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt\n// See:
https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\nimport kotlin.random.*\n\n/**\n * Returns a
random element from this range.\n * \n * @throws IllegalArgumentException if this range is empty.\n
*^@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline fun IntRange.random(): Int {\n    return
random(Random)\n}\n\n/**\n * Returns a random element from this range.\n * \n * @throws
IllegalArgumentException if this range is empty.\n *^@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic
inline fun LongRange.random(): Long {\n    return random(Random)\n}\n\n/**\n * Returns a random element from
this range.\n * \n * @throws IllegalArgumentException if this range is empty.\n
*^@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline fun CharRange.random(): Char {\n    return
random(Random)\n}\n\n/**\n * Returns a random element from this range using the specified source of
randomness.\n * \n * @throws IllegalArgumentException if this range is empty.\n
*^@SinceKotlin("1.3")\npublic fun IntRange.random(random: Random): Int {\n    try {\n        return
random.nextInt(this)\n    } catch (e: IllegalArgumentException) {\n        throw
NoSuchElementException(e.message)\n    }\n}\n\n/**\n * Returns a random element from this range using the
specified source of randomness.\n * \n * @throws IllegalArgumentException if this range is empty.\n
*^@SinceKotlin("1.3")\npublic fun LongRange.random(random: Random): Long {\n    try {\n        return
random.nextLong(this)\n    } catch (e: IllegalArgumentException) {\n        throw
NoSuchElementException(e.message)\n    }\n}\n\n/**\n * Returns a random element from this range using the
specified source of randomness.\n * \n * @throws IllegalArgumentException if this range is empty.\n
*^@SinceKotlin("1.3")\npublic fun CharRange.random(random: Random): Char {\n    try {\n        return
random.nextInt(first.code, last.code + 1).toChar()\n    } catch (e: IllegalArgumentException) {\n        throw

```

```

NoSuchElementException(e.message)}\n}\n\n/**\n * Returns a random element from this range, or `null` if this
range is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun IntRange.randomOrNull(): Int? {\n    return randomOrNull(Random)\n}\n\n/**\n * Returns a random
element from this range, or `null` if this range is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun LongRange.randomOrNull(): Long? {\n    return randomOrNull(Random)\n}\n\n/**\n * Returns a
random element from this range, or `null` if this range is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun CharRange.randomOrNull(): Char? {\n    return randomOrNull(Random)\n}\n\n/**\n * Returns a
random element from this range using the specified source of randomness, or `null` if this range is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
IntRange.randomOrNull(random: Random): Int? {\n    if (isEmpty())\n        return null\n    return
random.nextInt(this)\n}\n\n/**\n * Returns a random element from this range using the specified source of
randomness, or `null` if this range is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
LongRange.randomOrNull(random: Random): Long? {\n    if (isEmpty())\n        return null\n    return
random.nextLong(this)\n}\n\n/**\n * Returns a random element from this range using the specified source of
randomness, or `null` if this range is empty.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
CharRange.randomOrNull(random: Random): Char? {\n    if (isEmpty())\n        return null\n    return
random.nextInt(first.code, last.code + 1).toChar()\n}\n\n/**\n * Returns `true` if this range contains the specified
[element].\n * \n * Always returns `false` if the [element] is `null`.\n
*\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline operator fun IntRange.contains(element:
Int?): Boolean {\n    return element != null && contains(element)\n}\n\n/**\n * Returns `true` if this range contains
the specified [element].\n * \n * Always returns `false` if the [element] is `null`.\n
*\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline operator fun LongRange.contains(element:
Long?): Boolean {\n    return element != null && contains(element)\n}\n\n/**\n * Returns `true` if this range
contains the specified [element].\n * \n * Always returns `false` if the [element] is `null`.\n
*\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline operator fun CharRange.contains(element:
Char?): Boolean {\n    return element != null && contains(element)\n}\n\n/**\n * Checks if the specified [value]
belongs to this range.\n *\n@kotlin.jvm.JvmName("intRangeContains")\npublic operator fun
ClosedRange<Int>.contains(value: Byte): Boolean {\n    return contains(value.toInt())\n}\n\n/**\n * Checks if the
specified [value] belongs to this range.\n *\n@kotlin.jvm.JvmName("longRangeContains")\npublic operator fun
ClosedRange<Long>.contains(value: Byte): Boolean {\n    return contains(value.toLong())\n}\n\n/**\n * Checks if
the specified [value] belongs to this range.\n *\n@kotlin.jvm.JvmName("shortRangeContains")\npublic operator
fun ClosedRange<Short>.contains(value: Byte): Boolean {\n    return contains(value.toShort())\n}\n\n/**\n *
Checks if the specified [value] belongs to this range.\n *\n@Deprecated("This `contains` operation mixing integer
and floating point arguments has ambiguous semantics and is going to be
removed.")\n@DeprecatedSinceKotlin(warningSince = "1.3", errorSince = "1.4", hiddenSince =
"1.5")\n@kotlin.jvm.JvmName("doubleRangeContains")\npublic operator fun
ClosedRange<Double>.contains(value: Byte): Boolean {\n    return contains(value.toDouble())\n}\n\n/**\n *
Checks if the specified [value] belongs to this range.\n *\n@Deprecated("This `contains` operation mixing integer
and floating point arguments has ambiguous semantics and is going to be
removed.")\n@DeprecatedSinceKotlin(warningSince = "1.3", errorSince = "1.4", hiddenSince =
"1.5")\n@kotlin.jvm.JvmName("floatRangeContains")\npublic operator fun ClosedRange<Float>.contains(value:
Byte): Boolean {\n    return contains(value.toFloat())\n}\n\n/**\n * Checks if the specified [value] belongs to this
range.\n *\n@Deprecated("This `contains` operation mixing integer and floating point arguments has ambiguous

```

```

semantics and is going to be removed.}\n\n@DeprecatedSinceKotlin(warningSince = \"1.3\", errorSince = \"1.4\",
hiddenSince = \"1.5\")\n\n@kotlin.jvm.JvmName(\"intRangeContains\")\n\npublic operator fun
ClosedRange<Int>.contains(value: Double): Boolean {\n    return value.toIntExactOrNull().let { if (it != null)
contains(it) else false } }\n\n}\n\n/**\n * Checks if the specified [value] belongs to this range.\n
*\n\n@Deprecated(\"This `contains` operation mixing integer and floating point arguments has ambiguous semantics
and is going to be removed.}\n\n@DeprecatedSinceKotlin(warningSince = \"1.3\", errorSince = \"1.4\", hiddenSince
= \"1.5\")\n\n@kotlin.jvm.JvmName(\"longRangeContains\")\n\npublic operator fun
ClosedRange<Long>.contains(value: Double): Boolean {\n    return value.toLongExactOrNull().let { if (it != null)
contains(it) else false } }\n\n}\n\n/**\n * Checks if the specified [value] belongs to this range.\n
*\n\n@Deprecated(\"This `contains` operation mixing integer and floating point arguments has ambiguous semantics
and is going to be removed.}\n\n@DeprecatedSinceKotlin(warningSince = \"1.3\", errorSince = \"1.4\", hiddenSince
= \"1.5\")\n\n@kotlin.jvm.JvmName(\"byteRangeContains\")\n\npublic operator fun
ClosedRange<Byte>.contains(value: Double): Boolean {\n    return value.toByteExactOrNull().let { if (it != null)
contains(it) else false } }\n\n}\n\n/**\n * Checks if the specified [value] belongs to this range.\n
*\n\n@Deprecated(\"This `contains` operation mixing integer and floating point arguments has ambiguous semantics
and is going to be removed.}\n\n@DeprecatedSinceKotlin(warningSince = \"1.3\", errorSince = \"1.4\", hiddenSince
= \"1.5\")\n\n@kotlin.jvm.JvmName(\"shortRangeContains\")\n\npublic operator fun
ClosedRange<Short>.contains(value: Double): Boolean {\n    return value.toShortExactOrNull().let { if (it != null)
contains(it) else false } }\n\n}\n\n/**\n * Checks if the specified [value] belongs to this range.\n
*\n\n@kotlin.jvm.JvmName(\"floatRangeContains\")\n\npublic operator fun ClosedRange<Float>.contains(value:
Double): Boolean {\n    return contains(value.toFloat()) }\n\n}\n\n/**\n * Checks if the specified [value] belongs to this
range.\n *\n\n@Deprecated(\"This `contains` operation mixing integer and floating point arguments has ambiguous
semantics and is going to be removed.}\n\n@DeprecatedSinceKotlin(warningSince = \"1.3\", errorSince = \"1.4\",
hiddenSince = \"1.5\")\n\n@kotlin.jvm.JvmName(\"intRangeContains\")\n\npublic operator fun
ClosedRange<Int>.contains(value: Float): Boolean {\n    return value.toIntExactOrNull().let { if (it != null)
contains(it) else false } }\n\n}\n\n/**\n * Checks if the specified [value] belongs to this range.\n
*\n\n@Deprecated(\"This `contains` operation mixing integer and floating point arguments has ambiguous semantics
and is going to be removed.}\n\n@DeprecatedSinceKotlin(warningSince = \"1.3\", errorSince = \"1.4\", hiddenSince
= \"1.5\")\n\n@kotlin.jvm.JvmName(\"longRangeContains\")\n\npublic operator fun
ClosedRange<Long>.contains(value: Float): Boolean {\n    return value.toLongExactOrNull().let { if (it != null)
contains(it) else false } }\n\n}\n\n/**\n * Checks if the specified [value] belongs to this range.\n
*\n\n@Deprecated(\"This `contains` operation mixing integer and floating point arguments has ambiguous semantics
and is going to be removed.}\n\n@DeprecatedSinceKotlin(warningSince = \"1.3\", errorSince = \"1.4\", hiddenSince
= \"1.5\")\n\n@kotlin.jvm.JvmName(\"byteRangeContains\")\n\npublic operator fun
ClosedRange<Byte>.contains(value: Float): Boolean {\n    return value.toByteExactOrNull().let { if (it != null)
contains(it) else false } }\n\n}\n\n/**\n * Checks if the specified [value] belongs to this range.\n
*\n\n@Deprecated(\"This `contains` operation mixing integer and floating point arguments has ambiguous semantics
and is going to be removed.}\n\n@DeprecatedSinceKotlin(warningSince = \"1.3\", errorSince = \"1.4\", hiddenSince
= \"1.5\")\n\n@kotlin.jvm.JvmName(\"shortRangeContains\")\n\npublic operator fun
ClosedRange<Short>.contains(value: Float): Boolean {\n    return value.toShortExactOrNull().let { if (it != null)
contains(it) else false } }\n\n}\n\n/**\n * Checks if the specified [value] belongs to this range.\n
*\n\n@kotlin.jvm.JvmName(\"doubleRangeContains\")\n\npublic operator fun ClosedRange<Double>.contains(value:
Float): Boolean {\n    return contains(value.toDouble()) }\n\n}\n\n/**\n * Checks if the specified [value] belongs to this
range.\n *\n\n@kotlin.jvm.JvmName(\"longRangeContains\")\n\npublic operator fun
ClosedRange<Long>.contains(value: Int): Boolean {\n    return contains(value.toLong()) }\n\n}\n\n/**\n * Checks if the
specified [value] belongs to this range.\n *\n\n@kotlin.jvm.JvmName(\"byteRangeContains\")\n\npublic operator fun
ClosedRange<Byte>.contains(value: Int): Boolean {\n    return value.toByteExactOrNull().let { if (it != null)
contains(it) else false } }\n\n}\n\n/**\n * Checks if the specified [value] belongs to this range.\n

```

```

*@kotlin.jvm.JvmName("shortRangeContains")public operator fun ClosedRange<Short>.contains(value: Int): Boolean {\n return value.toShortExactOrNull().let { if (it != null) contains(it) else false }\n\n/**\n * Checks if the specified [value] belongs to this range.\n *\n@Deprecated("This `contains` operation mixing integer and floating point arguments has ambiguous semantics and is going to be removed.")\n@DeprecatedSinceKotlin(warningSince = "1.3", errorSince = "1.4", hiddenSince = "1.5")\n@kotlin.jvm.JvmName("doubleRangeContains")\npublic operator fun ClosedRange<Double>.contains(value: Int): Boolean {\n return contains(value.toDouble())\n\n/**\n * Checks if the specified [value] belongs to this range.\n *\n@Deprecated("This `contains` operation mixing integer and floating point arguments has ambiguous semantics and is going to be removed.")\n@DeprecatedSinceKotlin(warningSince = "1.3", errorSince = "1.4", hiddenSince = "1.5")\n@kotlin.jvm.JvmName("floatRangeContains")\npublic operator fun ClosedRange<Float>.contains(value: Int): Boolean {\n return contains(value.toFloat())\n\n/**\n * Checks if the specified [value] belongs to this range.\n *\n@kotlin.jvm.JvmName("intRangeContains")\npublic operator fun ClosedRange<Int>.contains(value: Long): Boolean {\n return value.toIntExactOrNull().let { if (it != null) contains(it) else false }\n\n/**\n * Checks if the specified [value] belongs to this range.\n *\n@kotlin.jvm.JvmName("byteRangeContains")\npublic operator fun ClosedRange<Byte>.contains(value: Long): Boolean {\n return value.toByteExactOrNull().let { if (it != null) contains(it) else false }\n\n/**\n * Checks if the specified [value] belongs to this range.\n *\n@kotlin.jvm.JvmName("shortRangeContains")\npublic operator fun ClosedRange<Short>.contains(value: Long): Boolean {\n return value.toShortExactOrNull().let { if (it != null) contains(it) else false }\n\n/**\n * Checks if the specified [value] belongs to this range.\n *\n@Deprecated("This `contains` operation mixing integer and floating point arguments has ambiguous semantics and is going to be removed.")\n@DeprecatedSinceKotlin(warningSince = "1.3", errorSince = "1.4", hiddenSince = "1.5")\n@kotlin.jvm.JvmName("doubleRangeContains")\npublic operator fun ClosedRange<Double>.contains(value: Long): Boolean {\n return contains(value.toDouble())\n\n/**\n * Checks if the specified [value] belongs to this range.\n *\n@Deprecated("This `contains` operation mixing integer and floating point arguments has ambiguous semantics and is going to be removed.")\n@DeprecatedSinceKotlin(warningSince = "1.3", errorSince = "1.4", hiddenSince = "1.5")\n@kotlin.jvm.JvmName("floatRangeContains")\npublic operator fun ClosedRange<Float>.contains(value: Long): Boolean {\n return contains(value.toFloat())\n\n/**\n * Checks if the specified [value] belongs to this range.\n *\n@kotlin.jvm.JvmName("intRangeContains")\npublic operator fun ClosedRange<Int>.contains(value: Short): Boolean {\n return contains(value.toInt())\n\n/**\n * Checks if the specified [value] belongs to this range.\n *\n@kotlin.jvm.JvmName("longRangeContains")\npublic operator fun ClosedRange<Long>.contains(value: Short): Boolean {\n return contains(value.toLong())\n\n/**\n * Checks if the specified [value] belongs to this range.\n *\n@kotlin.jvm.JvmName("byteRangeContains")\npublic operator fun ClosedRange<Byte>.contains(value: Short): Boolean {\n return value.toByteExactOrNull().let { if (it != null) contains(it) else false }\n\n/**\n * Checks if the specified [value] belongs to this range.\n *\n@Deprecated("This `contains` operation mixing integer and floating point arguments has ambiguous semantics and is going to be removed.")\n@DeprecatedSinceKotlin(warningSince = "1.3", errorSince = "1.4", hiddenSince = "1.5")\n@kotlin.jvm.JvmName("doubleRangeContains")\npublic operator fun ClosedRange<Double>.contains(value: Short): Boolean {\n return contains(value.toDouble())\n\n/**\n * Checks if the specified [value] belongs to this range.\n *\n@Deprecated("This `contains` operation mixing integer and floating point arguments has ambiguous semantics and is going to be removed.")\n@DeprecatedSinceKotlin(warningSince = "1.3", errorSince = "1.4", hiddenSince = "1.5")\n@kotlin.jvm.JvmName("floatRangeContains")\npublic operator fun ClosedRange<Float>.contains(value: Short): Boolean {\n return contains(value.toFloat())\n\n/**\n * Returns a progression from this value down to the specified [to] value with the step -1.\n * \n * The [to] value should be less than or equal to `this` value.\n * If the [to] value is greater than `this` value the returned progression is empty.\n *\n@public infix fun Int.downTo(to: Byte): IntProgression {\n return IntProgression.fromClosedRange(this, to.toInt(), -1)\n\n/**\n * Returns a progression

```



```

1.\n * \n * The [to] value should be less than or equal to `this` value.\n * If the [to] value is greater than `this` value
the returned progression is empty.\n */\npublic infix fun Byte.downTo(to: Short): IntProgression {\n    return
IntProgression.fromClosedRange(this.toInt(), to.toInt(), -1)\n}\n\n/**\n * Returns a progression from this value
down to the specified [to] value with the step -1.\n * \n * The [to] value should be less than or equal to `this` value.\n
* If the [to] value is greater than `this` value the returned progression is empty.\n */\npublic infix fun
Short.downTo(to: Short): IntProgression {\n    return IntProgression.fromClosedRange(this.toInt(), to.toInt(), -
1)\n}\n\n/**\n * Returns a progression that goes over the same range in the opposite direction with the same step.\n
*/\npublic fun IntProgression.reversed(): IntProgression {\n    return IntProgression.fromClosedRange(last, first, -
step)\n}\n\n/**\n * Returns a progression that goes over the same range in the opposite direction with the same
step.\n */\npublic fun LongProgression.reversed(): LongProgression {\n    return
LongProgression.fromClosedRange(last, first, -step)\n}\n\n/**\n * Returns a progression that goes over the same
range in the opposite direction with the same step.\n */\npublic fun CharProgression.reversed(): CharProgression {\n
    return CharProgression.fromClosedRange(last, first, -step)\n}\n\n/**\n * Returns a progression that goes over the
same range with the given step.\n */\npublic infix fun IntProgression.step(step: Int): IntProgression {\n
    checkStepIsPositive(step > 0, step)\n    return IntProgression.fromClosedRange(first, last, if (this.step > 0) step else -
step)\n}\n\n/**\n * Returns a progression that goes over the same range with the given step.\n */\npublic infix fun
LongProgression.step(step: Long): LongProgression {\n    checkStepIsPositive(step > 0, step)\n    return
LongProgression.fromClosedRange(first, last, if (this.step > 0) step else -step)\n}\n\n/**\n * Returns a progression
that goes over the same range with the given step.\n */\npublic infix fun CharProgression.step(step: Int):
CharProgression {\n    checkStepIsPositive(step > 0, step)\n    return CharProgression.fromClosedRange(first, last, if
(this.step > 0) step else -step)\n}\n\ninternal fun Int.toByteExactOrNull(): Byte? {\n    return if (this in
Byte.MIN_VALUE.toInt()..Byte.MAX_VALUE.toInt()) this.toByte() else null\n}\n\ninternal fun
Long.toByteExactOrNull(): Byte? {\n    return if (this in
Byte.MIN_VALUE.toLong()..Byte.MAX_VALUE.toLong()) this.toByte() else null\n}\n\ninternal fun
Short.toByteExactOrNull(): Byte? {\n    return if (this in
Byte.MIN_VALUE.toShort()..Byte.MAX_VALUE.toShort()) this.toByte() else null\n}\n\ninternal fun
Double.toByteExactOrNull(): Byte? {\n    return if (this in
Byte.MIN_VALUE.toDouble()..Byte.MAX_VALUE.toDouble()) this.toInt().toByte() else null\n}\n\ninternal fun
Float.toByteExactOrNull(): Byte? {\n    return if (this in
Byte.MIN_VALUE.toFloat()..Byte.MAX_VALUE.toFloat()) this.toInt().toByte() else null\n}\n\ninternal fun
Long.toIntExactOrNull(): Int? {\n    return if (this in Int.MIN_VALUE.toLong()..Int.MAX_VALUE.toLong())
this.toInt() else null\n}\n\ninternal fun Double.toIntExactOrNull(): Int? {\n    return if (this in
Int.MIN_VALUE.toDouble()..Int.MAX_VALUE.toDouble()) this.toInt() else null\n}\n\ninternal fun
Float.toIntExactOrNull(): Int? {\n    return if (this in Int.MIN_VALUE.toFloat()..Int.MAX_VALUE.toFloat())
this.toInt() else null\n}\n\ninternal fun Double.toLongExactOrNull(): Long? {\n    return if (this in
Long.MIN_VALUE.toDouble()..Long.MAX_VALUE.toDouble()) this.toLong() else null\n}\n\ninternal fun
Float.toLongExactOrNull(): Long? {\n    return if (this in
Long.MIN_VALUE.toFloat()..Long.MAX_VALUE.toFloat()) this.toLong() else null\n}\n\ninternal fun
Int.toShortExactOrNull(): Short? {\n    return if (this in Short.MIN_VALUE.toInt()..Short.MAX_VALUE.toInt())
this.toShort() else null\n}\n\ninternal fun Long.toShortExactOrNull(): Short? {\n    return if (this in
Short.MIN_VALUE.toLong()..Short.MAX_VALUE.toLong()) this.toShort() else null\n}\n\ninternal fun
Double.toShortExactOrNull(): Short? {\n    return if (this in
Short.MIN_VALUE.toDouble()..Short.MAX_VALUE.toDouble()) this.toInt().toShort() else null\n}\n\ninternal fun
Float.toShortExactOrNull(): Short? {\n    return if (this in
Short.MIN_VALUE.toFloat()..Short.MAX_VALUE.toFloat()) this.toInt().toShort() else null\n}\n\n/**\n * Returns
a range from this value up to but excluding the specified [to] value.\n * \n * If the [to] value is less than or equal to
`this` value, then the returned range is empty.\n */\npublic infix fun Int.until(to: Byte): IntRange {\n    return this ..
(to.toInt() - 1).toInt()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n
*

```

`\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Long.until(to: Byte): LongRange {\n return this .. (to.toLong() - 1).toLong()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Byte.until(to: Byte): IntRange {\n return this.toInt() .. (to.toInt() - 1).toInt()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Short.until(to: Byte): IntRange {\n return this.toInt() .. (to.toInt() - 1).toInt()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Char.until(to: Char): CharRange {\n if (to <= '\u0000')\n return CharRange.EMPTY\n return this .. (to - 1).toChar()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Int.until(to: Int): IntRange {\n if (to <= Int.MIN_VALUE) return IntRange.EMPTY\n return this .. (to - 1).toInt()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Long.until(to: Int): LongRange {\n return this .. (to.toLong() - 1).toLong()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Byte.until(to: Int): IntRange {\n if (to <= Int.MIN_VALUE) return IntRange.EMPTY\n return this.toInt() .. (to - 1).toInt()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Short.until(to: Int): IntRange {\n if (to <= Int.MIN_VALUE) return IntRange.EMPTY\n return this.toInt() .. (to - 1).toInt()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Int.until(to: Long): LongRange {\n if (to <= Long.MIN_VALUE) return LongRange.EMPTY\n return this.toLong() .. (to - 1).toLong()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Long.until(to: Long): LongRange {\n if (to <= Long.MIN_VALUE) return LongRange.EMPTY\n return this .. (to - 1).toLong()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Byte.until(to: Long): LongRange {\n if (to <= Long.MIN_VALUE) return LongRange.EMPTY\n return this.toLong() .. (to - 1).toLong()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Short.until(to: Long): LongRange {\n if (to <= Long.MIN_VALUE) return LongRange.EMPTY\n return this.toLong() .. (to - 1).toLong()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Int.until(to: Short): IntRange {\n return this .. (to.toInt() - 1).toInt()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Long.until(to: Short): LongRange {\n return this .. (to.toLong() - 1).toLong()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Byte.until(to: Short): IntRange {\n return this.toInt() .. (to.toInt() - 1).toInt()\n}\n\n/**\n * Returns a range from this value up to but excluding the specified [to] value.\n *\n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n */\npublic infix fun Short.until(to: Short): IntRange {\n return this.toInt() .. (to.toInt() - 1).toInt()\n}\n}\n\n/**\n * Ensures that this value is not less than the specified [minimumValue].\n *\n * @return this value if it's greater than or equal to the [minimumValue] or the [minimumValue] otherwise.\n *\n * @sample\n samples.comparisons.ComparableOps.coerceAtLeastComparable\n */\npublic fun <T : Comparable<T>> T.coerceAtLeast(minimumValue: T): T {\n return if (this < minimumValue) minimumValue else this\n}\n\n/**\n * Ensures that this value is not less than the specified [minimumValue].\n *\n * @return this value if it's greater`

```

than or equal to the [minimumValue] or the [minimumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtLeast\n *^\npublic fun Byte.coerceAtLeast(minimumValue: Byte):
Byte {\n    return if (this < minimumValue) minimumValue else this\n}\n\n/**\n * Ensures that this value is not less
than the specified [minimumValue].\n * \n * @return this value if it's greater than or equal to the [minimumValue]
or the [minimumValue] otherwise.\n * \n * @sample samples.comparisons.ComparableOps.coerceAtLeast\n
*^\npublic fun Short.coerceAtLeast(minimumValue: Short): Short {\n    return if (this < minimumValue)
minimumValue else this\n}\n\n/**\n * Ensures that this value is not less than the specified [minimumValue].\n * \n
* @return this value if it's greater than or equal to the [minimumValue] or the [minimumValue] otherwise.\n * \n *
@sample samples.comparisons.ComparableOps.coerceAtLeast\n *^\npublic fun Int.coerceAtLeast(minimumValue:
Int): Int {\n    return if (this < minimumValue) minimumValue else this\n}\n\n/**\n * Ensures that this value is not
less than the specified [minimumValue].\n * \n * @return this value if it's greater than or equal to the
[minimumValue] or the [minimumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtLeast\n *^\npublic fun Long.coerceAtLeast(minimumValue: Long):
Long {\n    return if (this < minimumValue) minimumValue else this\n}\n\n/**\n * Ensures that this value is not less
than the specified [minimumValue].\n * \n * @return this value if it's greater than or equal to the [minimumValue]
or the [minimumValue] otherwise.\n * \n * @sample samples.comparisons.ComparableOps.coerceAtLeast\n
*^\npublic fun Float.coerceAtLeast(minimumValue: Float): Float {\n    return if (this < minimumValue)
minimumValue else this\n}\n\n/**\n * Ensures that this value is not less than the specified [minimumValue].\n * \n
* @return this value if it's greater than or equal to the [minimumValue] or the [minimumValue] otherwise.\n * \n *
@sample samples.comparisons.ComparableOps.coerceAtLeast\n *^\npublic fun
Double.coerceAtLeast(minimumValue: Double): Double {\n    return if (this < minimumValue) minimumValue else
this\n}\n\n/**\n * Ensures that this value is not greater than the specified [maximumValue].\n * \n * @return this
value if it's less than or equal to the [maximumValue] or the [maximumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtMostComparable\n *^\npublic fun <T : Comparable<T>>
T.coerceAtMost(maximumValue: T): T {\n    return if (this > maximumValue) maximumValue else this\n}\n\n/**\n * Ensures that this value is not greater than the specified [maximumValue].\n * \n * @return this value if it's less
than or equal to the [maximumValue] or the [maximumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtMost\n *^\npublic fun Byte.coerceAtMost(maximumValue: Byte):
Byte {\n    return if (this > maximumValue) maximumValue else this\n}\n\n/**\n * Ensures that this value is not
greater than the specified [maximumValue].\n * \n * @return this value if it's less than or equal to the
[maximumValue] or the [maximumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtMost\n *^\npublic fun Short.coerceAtMost(maximumValue: Short):
Short {\n    return if (this > maximumValue) maximumValue else this\n}\n\n/**\n * Ensures that this value is not
greater than the specified [maximumValue].\n * \n * @return this value if it's less than or equal to the
[maximumValue] or the [maximumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtMost\n *^\npublic fun Int.coerceAtMost(maximumValue: Int): Int
{\n    return if (this > maximumValue) maximumValue else this\n}\n\n/**\n * Ensures that this value is not greater
than the specified [maximumValue].\n * \n * @return this value if it's less than or equal to the [maximumValue] or
the [maximumValue] otherwise.\n * \n * @sample samples.comparisons.ComparableOps.coerceAtMost\n
*^\npublic fun Long.coerceAtMost(maximumValue: Long): Long {\n    return if (this > maximumValue)
maximumValue else this\n}\n\n/**\n * Ensures that this value is not greater than the specified [maximumValue].\n * \n
* @return this value if it's less than or equal to the [maximumValue] or the [maximumValue] otherwise.\n * \n *
@sample samples.comparisons.ComparableOps.coerceAtMost\n *^\npublic fun
Float.coerceAtMost(maximumValue: Float): Float {\n    return if (this > maximumValue) maximumValue else
this\n}\n\n/**\n * Ensures that this value is not greater than the specified [maximumValue].\n * \n * @return this
value if it's less than or equal to the [maximumValue] or the [maximumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtMost\n *^\npublic fun Double.coerceAtMost(maximumValue:
Double): Double {\n    return if (this > maximumValue) maximumValue else this\n}\n\n/**\n * Ensures that this

```

```

value lies in the specified range [minimumValue]..[maximumValue].\n * \n * @return this value if it's in the range,
or [minimumValue] if this value is less than [minimumValue], or [maximumValue] if this value is greater than
[maximumValue].\n * \n * @sample samples.comparisons.ComparableOps.coerceInComparable\n *^\npublic fun
<T : Comparable<T>> T.coerceIn(minimumValue: T?, maximumValue: T?): T {\n if (minimumValue !== null
&& maximumValue !== null) {\n if (minimumValue > maximumValue) throw
IllegalArgumentException("Cannot coerce value to an empty range: maximum $maximumValue is less than
minimum $minimumValue.")\n if (this < minimumValue) return minimumValue\n if (this >
maximumValue) return maximumValue\n } \n else {\n if (minimumValue !== null && this <
minimumValue) return minimumValue\n if (maximumValue !== null && this > maximumValue) return
maximumValue\n } \n return this\n}\n\n/**\n * Ensures that this value lies in the specified range
[minimumValue]..[maximumValue].\n * \n * @return this value if it's in the range, or [minimumValue] if this value
is less than [minimumValue], or [maximumValue] if this value is greater than [maximumValue].\n * \n * @sample
samples.comparisons.ComparableOps.coerceIn\n *^\npublic fun Byte.coerceIn(minimumValue: Byte,
maximumValue: Byte): Byte {\n if (minimumValue > maximumValue) throw
IllegalArgumentException("Cannot coerce value to an empty range: maximum $maximumValue is less than
minimum $minimumValue.")\n if (this < minimumValue) return minimumValue\n if (this > maximumValue)
return maximumValue\n return this\n}\n\n/**\n * Ensures that this value lies in the specified range
[minimumValue]..[maximumValue].\n * \n * @return this value if it's in the range, or [minimumValue] if this value
is less than [minimumValue], or [maximumValue] if this value is greater than [maximumValue].\n * \n * @sample
samples.comparisons.ComparableOps.coerceIn\n *^\npublic fun Short.coerceIn(minimumValue: Short,
maximumValue: Short): Short {\n if (minimumValue > maximumValue) throw
IllegalArgumentException("Cannot coerce value to an empty range: maximum $maximumValue is less than
minimum $minimumValue.")\n if (this < minimumValue) return minimumValue\n if (this > maximumValue)
return maximumValue\n return this\n}\n\n/**\n * Ensures that this value lies in the specified range
[minimumValue]..[maximumValue].\n * \n * @return this value if it's in the range, or [minimumValue] if this value
is less than [minimumValue], or [maximumValue] if this value is greater than [maximumValue].\n * \n * @sample
samples.comparisons.ComparableOps.coerceIn\n *^\npublic fun Int.coerceIn(minimumValue: Int, maximumValue:
Int): Int {\n if (minimumValue > maximumValue) throw IllegalArgumentException("Cannot coerce value to an
empty range: maximum $maximumValue is less than minimum $minimumValue.")\n if (this < minimumValue)
return minimumValue\n if (this > maximumValue) return maximumValue\n return this\n}\n\n/**\n * Ensures
that this value lies in the specified range [minimumValue]..[maximumValue].\n * \n * @return this value if it's in
the range, or [minimumValue] if this value is less than [minimumValue], or [maximumValue] if this value is greater
than [maximumValue].\n * \n * @sample samples.comparisons.ComparableOps.coerceIn\n *^\npublic fun
Long.coerceIn(minimumValue: Long, maximumValue: Long): Long {\n if (minimumValue > maximumValue)
throw IllegalArgumentException("Cannot coerce value to an empty range: maximum $maximumValue is less than
minimum $minimumValue.")\n if (this < minimumValue) return minimumValue\n if (this > maximumValue)
return maximumValue\n return this\n}\n\n/**\n * Ensures that this value lies in the specified range
[minimumValue]..[maximumValue].\n * \n * @return this value if it's in the range, or [minimumValue] if this value
is less than [minimumValue], or [maximumValue] if this value is greater than [maximumValue].\n * \n * @sample
samples.comparisons.ComparableOps.coerceIn\n *^\npublic fun Float.coerceIn(minimumValue: Float,
maximumValue: Float): Float {\n if (minimumValue > maximumValue) throw
IllegalArgumentException("Cannot coerce value to an empty range: maximum $maximumValue is less than
minimum $minimumValue.")\n if (this < minimumValue) return minimumValue\n if (this > maximumValue)
return maximumValue\n return this\n}\n\n/**\n * Ensures that this value lies in the specified range
[minimumValue]..[maximumValue].\n * \n * @return this value if it's in the range, or [minimumValue] if this value
is less than [minimumValue], or [maximumValue] if this value is greater than [maximumValue].\n * \n * @sample
samples.comparisons.ComparableOps.coerceIn\n *^\npublic fun Double.coerceIn(minimumValue: Double,
maximumValue: Double): Double {\n if (minimumValue > maximumValue) throw

```

```

IllegalArgumentException("Cannot coerce value to an empty range: maximum $maximumValue is less than
minimum $minimumValue.")\n  if (this < minimumValue) return minimumValue\n  if (this > maximumValue)
return maximumValue\n  return this\n}\n\n/**\n * Ensures that this value lies in the specified [range].\n * \n *
@return this value if it's in the [range], or `range.start` if this value is less than `range.start`, or `range.endInclusive`
if this value is greater than `range.endInclusive`.\n * \n * @sample
samples.comparisons.ComparableOps.coerceInFloatingPointRange\n *^\n@SinceKotlin("1.1")\npublic fun <T :
Comparable<T>> T.coerceIn(range: ClosedFloatingPointRange<T>): T {\n  if (range.isEmpty()) throw
IllegalArgumentException("Cannot coerce value to an empty range: $range.")\n  return when {\n    // this <
start equiv to this <= start && !(this >= start)\n    range.lessThanOrEqualTo(this, range.start) &&
!range.lessThanOrEqualTo(range.start, this) -> range.start\n    // this > end equiv to this >= end && !(this <= end)\n
range.lessThanOrEqualTo(range.endInclusive, this) && !range.lessThanOrEqualTo(this, range.endInclusive) ->
range.endInclusive\n    else -> this\n  }\n}\n\n/**\n * Ensures that this value lies in the specified [range].\n * \n *
@return this value if it's in the [range], or `range.start` if this value is less than `range.start`, or `range.endInclusive`
if this value is greater than `range.endInclusive`.\n * \n * @sample
samples.comparisons.ComparableOps.coerceInComparable\n *^\npublic fun <T : Comparable<T>>
T.coerceIn(range: ClosedRange<T>): T {\n  if (range is ClosedFloatingPointRange) {\n    return
this.coerceIn<T>(range)\n  }\n  if (range.isEmpty()) throw IllegalArgumentException("Cannot coerce value to an
empty range: $range.")\n  return when {\n    this < range.start -> range.start\n    this > range.endInclusive ->
range.endInclusive\n    else -> this\n  }\n}\n\n/**\n * Ensures that this value lies in the specified [range].\n * \n *
@return this value if it's in the [range], or `range.start` if this value is less than `range.start`, or `range.endInclusive`
if this value is greater than `range.endInclusive`.\n * \n * @sample samples.comparisons.ComparableOps.coerceIn\n
*^\npublic fun Int.coerceIn(range: ClosedRange<Int>): Int {\n  if (range is ClosedFloatingPointRange) {\n
return this.coerceIn<Int>(range)\n  }\n  if (range.isEmpty()) throw IllegalArgumentException("Cannot coerce
value to an empty range: $range.")\n  return when {\n    this < range.start -> range.start\n    this >
range.endInclusive -> range.endInclusive\n    else -> this\n  }\n}\n\n/**\n * Ensures that this value lies in the
specified [range].\n * \n * @return this value if it's in the [range], or `range.start` if this value is less than
`range.start`, or `range.endInclusive` if this value is greater than `range.endInclusive`.\n * \n * @sample
samples.comparisons.ComparableOps.coerceIn\n *^\npublic fun Long.coerceIn(range: ClosedRange<Long>): Long
{\n  if (range is ClosedFloatingPointRange) {\n    return this.coerceIn<Long>(range)\n  }\n  if
(range.isEmpty()) throw IllegalArgumentException("Cannot coerce value to an empty range: $range.")\n  return
when {\n    this < range.start -> range.start\n    this > range.endInclusive -> range.endInclusive\n    else ->
this\n  }\n}\n\n"/\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n *
Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*/\n\n// Auto-generated file. DO NOT EDIT!\n\npackage kotlin\n\nimport kotlin.experimental.*\nimport
kotlin.jvm.*\n\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@JvmInline\npublic
value class UByte @PublishedApi internal constructor(@PublishedApi internal val data: Byte) :
Comparable<UByte> {\n  companion object {\n    /**\n     * A constant holding the minimum value an
instance of UByte can have.\n     */\n    public const val MIN_VALUE: UByte = UByte(0)\n\n    /**\n     *
A constant holding the maximum value an instance of UByte can have.\n     */\n    public const val
MAX_VALUE: UByte = UByte(-1)\n\n    /**\n     * The number of bytes used to represent an instance of
UByte in a binary form.\n     */\n    public const val SIZE_BYTES: Int = 1\n\n    /**\n     * The number of
bits used to represent an instance of UByte in a binary form.\n     */\n    public const val SIZE_BITS: Int = 8\n
}\n\n    /**\n     * Compares this value with the specified value for order.\n     * Returns zero if this value is equal to
the specified other value, a negative number if it's less than other,\n     * or a positive number if it's greater than
other.\n     */\n    @kotlin.internal.InlineOnly\n    @Suppress("OVERRIDE_BY_INLINE")\n    public override
inline operator fun compareTo(other: UByte): Int = this.toInt().compareTo(other.toInt())\n\n    /**\n     * Compares
this value with the specified value for order.\n     * Returns zero if this value is equal to the specified other value, a
negative number if it's less than other,\n     * or a positive number if it's greater than other.\n     */\n
}

```

```

@kotlin.internal.InlineOnly\n public inline operator fun compareTo(other: UShort): Int =
this.toInt().compareTo(other.toInt())\n\n /**\n * Compares this value with the specified value for order.\n *
Returns zero if this value is equal to the specified other value, a negative number if it's less than other,\n * or a
positive number if it's greater than other.\n */\n @kotlin.internal.InlineOnly\n public inline operator fun
compareTo(other: UInt): Int = this.toUInt().compareTo(other)\n\n /**\n * Compares this value with the
specified value for order.\n * Returns zero if this value is equal to the specified other value, a negative number if
it's less than other,\n * or a positive number if it's greater than other.\n */\n @kotlin.internal.InlineOnly\n
public inline operator fun compareTo(other: ULong): Int = this.toULong().compareTo(other)\n\n /** Adds the
other value to this value. */\n @kotlin.internal.InlineOnly\n public inline operator fun plus(other: UByte): UInt =
this.toUInt().plus(other.toUInt())\n\n /** Adds the other value to this value. */\n @kotlin.internal.InlineOnly\n
public inline operator fun plus(other: UShort): UInt = this.toUInt().plus(other.toUInt())\n\n /** Adds the other value
to this value. */\n @kotlin.internal.InlineOnly\n public inline operator fun plus(other: UInt): UInt =
this.toUInt().plus(other)\n\n /** Adds the other value to this value. */\n @kotlin.internal.InlineOnly\n public
inline operator fun plus(other: ULong): ULong = this.toULong().plus(other)\n\n /** Subtracts the other value from
this value. */\n @kotlin.internal.InlineOnly\n public inline operator fun minus(other: UByte): UInt =
this.toUInt().minus(other.toUInt())\n\n /** Subtracts the other value from this value. */\n
@kotlin.internal.InlineOnly\n public inline operator fun minus(other: UShort): UInt =
this.toUInt().minus(other.toUInt())\n\n /** Subtracts the other value from this value. */\n
@kotlin.internal.InlineOnly\n public inline operator fun minus(other: UInt): UInt = this.toUInt().minus(other)\n\n
/** Subtracts the other value from this value. */\n @kotlin.internal.InlineOnly\n public inline operator fun
minus(other: ULong): ULong = this.toULong().minus(other)\n\n /** Multiplies this value by the other value. */\n
@kotlin.internal.InlineOnly\n public inline operator fun times(other: UByte): UInt =
this.toUInt().times(other.toUInt())\n\n /** Multiplies this value by the other value. */\n
@kotlin.internal.InlineOnly\n public inline operator fun times(other: UShort): UInt =
this.toUInt().times(other.toUInt())\n\n /** Multiplies this value by the other value. */\n
@kotlin.internal.InlineOnly\n public inline operator fun times(other: UInt): UInt = this.toUInt().times(other)\n\n
/** Multiplies this value by the other value. */\n @kotlin.internal.InlineOnly\n public inline operator fun
times(other: ULong): ULong = this.toULong().times(other)\n\n /** Divides this value by the other value,
truncating the result to an integer that is closer to zero. */\n @kotlin.internal.InlineOnly\n public inline operator
fun div(other: UByte): UInt = this.toUInt().div(other.toUInt())\n\n /** Divides this value by the other value,
truncating the result to an integer that is closer to zero. */\n @kotlin.internal.InlineOnly\n public inline operator
fun div(other: UShort): UInt = this.toUInt().div(other.toUInt())\n\n /** Divides this value by the other value,
truncating the result to an integer that is closer to zero. */\n @kotlin.internal.InlineOnly\n public inline operator
fun div(other: UInt): UInt = this.toUInt().div(other)\n\n /** Divides this value by the other value, truncating the
result to an integer that is closer to zero. */\n @kotlin.internal.InlineOnly\n public inline operator fun div(other:
ULong): ULong = this.toULong().div(other)\n\n /**\n * Calculates the remainder of truncating division of this
value by the other value.\n * \n * The result is always less than the divisor.\n */\n
@kotlin.internal.InlineOnly\n public inline operator fun rem(other: UByte): UInt =
this.toUInt().rem(other.toUInt())\n\n /**\n * Calculates the remainder of truncating division of this value by the
other value.\n * \n * The result is always less than the divisor.\n */\n @kotlin.internal.InlineOnly\n public
inline operator fun rem(other: UShort): UInt = this.toUInt().rem(other.toUInt())\n\n /**\n * Calculates the
remainder of truncating division of this value by the other value.\n * \n * The result is always less than the
divisor.\n */\n @kotlin.internal.InlineOnly\n public inline operator fun rem(other: UInt): UInt =
this.toUInt().rem(other)\n\n /**\n * Calculates the remainder of truncating division of this value by the other
value.\n * \n * The result is always less than the divisor.\n */\n @kotlin.internal.InlineOnly\n public
inline operator fun rem(other: ULong): ULong = this.toULong().rem(other)\n\n /**\n * Divides this value by
the other value, flooring the result to an integer that is closer to negative infinity.\n * \n * For unsigned types,
the results of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n

```

```

public inline fun floorDiv(other: UByte): UInt = this.toUInt().floorDiv(other.toUInt())\n /**\n * Divides this
value by the other value, flooring the result to an integer that is closer to negative infinity.\n *\n * For unsigned
types, the results of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n
public inline fun floorDiv(other: UShort): UInt = this.toUInt().floorDiv(other.toUInt())\n /**\n * Divides this
value by the other value, flooring the result to an integer that is closer to negative infinity.\n *\n * For unsigned
types, the results of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n
public inline fun floorDiv(other: UInt): UInt = this.toUInt().floorDiv(other)\n /**\n * Divides this value by the
other value, flooring the result to an integer that is closer to negative infinity.\n *\n * For unsigned types, the
results of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n public
inline fun floorDiv(other: ULong): ULong = this.toULong().floorDiv(other)\n\n /**\n * Calculates the
remainder of flooring division of this value by the other value.\n *\n * The result is always less than the
divisor.\n *\n * For unsigned types, the remainders of flooring division and truncating division are the same.\n
*/\n @kotlin.internal.InlineOnly\n public inline fun mod(other: UByte): UByte =
this.toUInt().mod(other.toUInt()).toUByte()\n /**\n * Calculates the remainder of flooring division of this value
by the other value.\n *\n * The result is always less than the divisor.\n *\n * For unsigned types, the
remainders of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n
public inline fun mod(other: UShort): UShort = this.toUInt().mod(other.toUInt()).toUShort()\n /**\n *
Calculates the remainder of flooring division of this value by the other value.\n *\n * The result is always less
than the divisor.\n *\n * For unsigned types, the remainders of flooring division and truncating division are the
same.\n */\n @kotlin.internal.InlineOnly\n public inline fun mod(other: UInt): UInt =
this.toUInt().mod(other)\n /**\n * Calculates the remainder of flooring division of this value by the other
value.\n *\n * The result is always less than the divisor.\n *\n * For unsigned types, the remainders of
flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n public inline fun
mod(other: ULong): ULong = this.toULong().mod(other)\n\n /**\n * Returns this value incremented by one.\n
*/\n @sample samples.misc.Builtins.inc\n */\n @kotlin.internal.InlineOnly\n public inline operator fun
inc(): UByte = UByte(data.inc())\n\n /**\n * Returns this value decremented by one.\n */\n @sample
samples.misc.Builtins.dec\n */\n @kotlin.internal.InlineOnly\n public inline operator fun dec(): UByte =
UByte(data.dec())\n\n /** Creates a range from this value to the specified [other] value. */\n
@kotlin.internal.InlineOnly\n public inline operator fun rangeTo(other: UByte): UIntRange =
UIntRange(this.toUInt(), other.toUInt())\n\n /** Performs a bitwise AND operation between the two values. */\n
@kotlin.internal.InlineOnly\n public inline infix fun and(other: UByte): UByte = UByte(this.data and other.data)\n
\n /** Performs a bitwise OR operation between the two values. */\n @kotlin.internal.InlineOnly\n public inline
infix fun or(other: UByte): UByte = UByte(this.data or other.data)\n\n /** Performs a bitwise XOR operation
between the two values. */\n @kotlin.internal.InlineOnly\n public inline infix fun xor(other: UByte): UByte =
UByte(this.data xor other.data)\n\n /** Inverts the bits in this value. */\n @kotlin.internal.InlineOnly\n public
inline fun inv(): UByte = UByte(data.inv())\n\n /**\n * Converts this [UByte] value to [Byte].\n *\n * If
this value is less than or equals to [Byte.MAX_VALUE], the resulting `Byte` value represents\n * the same
numerical value as this `UByte`. Otherwise the result is negative.\n *\n * The resulting `Byte` value has the
same binary representation as this `UByte` value.\n */\n @kotlin.internal.InlineOnly\n public inline fun
toByte(): Byte = data\n\n /**\n * Converts this [UByte] value to [Short].\n *\n * The resulting `Short` value
represents the same numerical value as this `UByte`. \n *\n * The least significant 8 bits of the resulting `Short`
value are the same as the bits of this `UByte` value,\n * whereas the most significant 8 bits are filled with zeros.\n
*/\n @kotlin.internal.InlineOnly\n public inline fun toShort(): Short = data.toShort() and 0xFF\n\n /**\n *
Converts this [UByte] value to [Int].\n *\n * The resulting `Int` value represents the same numerical value as
this `UByte`. \n *\n * The least significant 8 bits of the resulting `Int` value are the same as the bits of this
`UByte` value,\n * whereas the most significant 24 bits are filled with zeros.\n */\n
@kotlin.internal.InlineOnly\n public inline fun toInt(): Int = data.toInt() and 0xFF\n\n /**\n * Converts this
[UByte] value to [Long].\n *\n * The resulting `Long` value represents the same numerical value as this

```

```

`UByte`.n *n * The least significant 8 bits of the resulting `Long` value are the same as the bits of this
`UByte` value,n * whereas the most significant 56 bits are filled with zeros.n */n
@kotlin.internal.InlineOnlyn public inline fun toLong(): Long = data.toLong() and 0xFFn/n /** Returns this
value. */n @kotlin.internal.InlineOnlyn public inline fun toUByte(): UByte = thisn /**n * Converts this
[UByte] value to [UShort].n *n * The resulting `UShort` value represents the same numerical value as this
`UByte`.n *n * The least significant 8 bits of the resulting `UShort` value are the same as the bits of this
`UByte` value,n * whereas the most significant 8 bits are filled with zeros.n */n
@kotlin.internal.InlineOnlyn public inline fun toUShort(): UShort = UShort(data.toShort() and 0xFF)n/n /**n
* Converts this [UByte] value to [UInt].n *n * The resulting `UInt` value represents the same numerical value
as this `UByte`.n *n * The least significant 8 bits of the resulting `UInt` value are the same as the bits of this
`UByte` value,n * whereas the most significant 24 bits are filled with zeros.n */n
@kotlin.internal.InlineOnlyn public inline fun toUInt(): UInt = UInt(data.toInt() and 0xFF)n/n /**n *
Converts this [UByte] value to [ULong].n *n * The resulting `ULong` value represents the same numerical
value as this `UByte`.n *n * The least significant 8 bits of the resulting `ULong` value are the same as the bits
of this `UByte` value,n * whereas the most significant 56 bits are filled with zeros.n */n
@kotlin.internal.InlineOnlyn public inline fun toULong(): ULong = ULong(data.toLong() and 0xFF)n/n /**n
* Converts this [UByte] value to [Float].n *n * The resulting `Float` value represents the same numerical
value as this `UByte`.n */n @kotlin.internal.InlineOnlyn public inline fun toFloat(): Float =
this.toInt().toFloat()n/n /**n * Converts this [UByte] value to [Double].n *n * The resulting `Double`
value represents the same numerical value as this `UByte`.n */n @kotlin.internal.InlineOnlyn public inline
fun toDouble(): Double = this.toInt().toDouble()n/n public override fun toString(): String =
toInt().toString()n/n}n/n/**n * Converts this [Byte] value to [UByte].n */n * If this value is positive, the resulting
`UByte` value represents the same numerical value as this `Byte`.n */n * The resulting `UByte` value has the same
binary representation as this `Byte` value.n
*/n@SinceKotlin("1.5")n@WasExperimental(ExperimentalUnsignedTypes::class)n@kotlin.internal.InlineOnlyn
npublic inline fun Byte.toUByte(): UByte = UByte(this)n/n/**n * Converts this [Short] value to [UByte].n */n * If
this value is positive and less than or equals to [UByte.MAX_VALUE], the resulting `UByte` value representsn *
the same numerical value as this `Short`.n */n * The resulting `UByte` value is represented by the least significant 8
bits of this `Short` value.n
*/n@SinceKotlin("1.5")n@WasExperimental(ExperimentalUnsignedTypes::class)n@kotlin.internal.InlineOnlyn
npublic inline fun Short.toUByte(): UByte = UByte(this.toByte())n/n/**n * Converts this [Int] value to [UByte].n
*/n * If this value is positive and less than or equals to [UByte.MAX_VALUE], the resulting `UByte` value
representsn * the same numerical value as this `Int`.n */n * The resulting `UByte` value is represented by the least
significant 8 bits of this `Int` value.n
*/n@SinceKotlin("1.5")n@WasExperimental(ExperimentalUnsignedTypes::class)n@kotlin.internal.InlineOnlyn
npublic inline fun Int.toUByte(): UByte = UByte(this.toByte())n/n/**n * Converts this [Long] value to [UByte].n
*/n * If this value is positive and less than or equals to [UByte.MAX_VALUE], the resulting `UByte` value
representsn * the same numerical value as this `Long`.n */n * The resulting `UByte` value is represented by the
least significant 8 bits of this `Long` value.n
*/n@SinceKotlin("1.5")n@WasExperimental(ExperimentalUnsignedTypes::class)n@kotlin.internal.InlineOnlyn
npublic inline fun Long.toUByte(): UByte = UByte(this.toByte())n"/n/**n * Copyright 2010-2021 JetBrains s.r.o.
and Kotlin Programming Language contributors.n * Use of this source code is governed by the Apache 2.0 license
that can be found in the license/LICENSE.txt file.n */n/n// Auto-generated file. DO NOT EDIT!n/npackage
kotlin\n\nimport kotlin.experimental.*nimport
kotlin.jvm.*n/n@SinceKotlin("1.5")n@WasExperimental(ExperimentalUnsignedTypes::class)n@JvmInline\npub
lic value class UInt @PublishedApi internal constructor(@PublishedApi internal val data: Int) :
Comparable<UInt> {n/n companion object {n/n /**n * A constant holding the minimum value an
instance of UInt can have.n */n public const val MIN_VALUE: UInt = UInt(0)n/n /**n * A

```

```

constant holding the maximum value an instance of UInt can have.\n    *^\n    public const val MAX_VALUE:
UInt = UInt(-1)\n\n    /**\n    * The number of bytes used to represent an instance of UInt in a binary form.\n    *^\n    public const val SIZE_BYTES: Int = 4\n\n    /**\n    * The number of bits used to represent an
instance of UInt in a binary form.\n    *^\n    public const val SIZE_BITS: Int = 32\n    }\n\n    /**\n    *
Compares this value with the specified value for order.\n    * Returns zero if this value is equal to the specified other
value, a negative number if it's less than other,\n    * or a positive number if it's greater than other.\n    *^\n
@kotlin.internal.InlineOnly\n    public inline operator fun compareTo(other: UByte): Int =
this.compareTo(other.toUInt())\n\n    /**\n    * Compares this value with the specified value for order.\n    *
Returns zero if this value is equal to the specified other value, a negative number if it's less than other,\n    * or a
positive number if it's greater than other.\n    *^\n    @kotlin.internal.InlineOnly\n    public inline operator fun
compareTo(other: UShort): Int = this.compareTo(other.toUInt())\n\n    /**\n    * Compares this value with the
specified value for order.\n    * Returns zero if this value is equal to the specified other value, a negative number if
it's less than other,\n    * or a positive number if it's greater than other.\n    *^\n    @kotlin.internal.InlineOnly\n
@Suppress("OVERRIDE_BY_INLINE")\n    public override inline operator fun compareTo(other: UInt): Int =
uintCompare(this.data, other.data)\n\n    /**\n    * Compares this value with the specified value for order.\n    *
Returns zero if this value is equal to the specified other value, a negative number if it's less than other,\n    * or a
positive number if it's greater than other.\n    *^\n    @kotlin.internal.InlineOnly\n    public inline operator fun
compareTo(other: ULong): Int = this.toULong().compareTo(other)\n\n    /** Adds the other value to this value. *^\n
    @kotlin.internal.InlineOnly\n    public inline operator fun plus(other: UByte): UInt = this.plus(other.toUInt())\n
/** Adds the other value to this value. *^\n    @kotlin.internal.InlineOnly\n    public inline operator fun plus(other:
UShort): UInt = this.plus(other.toUInt())\n    /** Adds the other value to this value. *^\n
    @kotlin.internal.InlineOnly\n    public inline operator fun plus(other: UInt): UInt = UInt(this.data.plus(other.data))\n
/** Adds the other value to this value. *^\n    @kotlin.internal.InlineOnly\n    public inline operator fun plus(other:
ULong): ULong = this.toULong().plus(other)\n\n    /** Subtracts the other value from this value. *^\n
    @kotlin.internal.InlineOnly\n    public inline operator fun minus(other: UByte): UInt = this.minus(other.toUInt())\n
/** Subtracts the other value from this value. *^\n    @kotlin.internal.InlineOnly\n    public inline operator fun
minus(other: UShort): UInt = this.minus(other.toUInt())\n    /** Subtracts the other value from this value. *^\n
    @kotlin.internal.InlineOnly\n    public inline operator fun minus(other: UInt): UInt =
UInt(this.data.minus(other.data))\n    /** Subtracts the other value from this value. *^\n
    @kotlin.internal.InlineOnly\n    public inline operator fun minus(other: ULong): ULong =
this.toULong().minus(other)\n\n    /** Multiplies this value by the other value. *^\n    @kotlin.internal.InlineOnly\n
public inline operator fun times(other: UByte): UInt = this.times(other.toUInt())\n    /** Multiplies this value by the
other value. *^\n    @kotlin.internal.InlineOnly\n    public inline operator fun times(other: UShort): UInt =
this.times(other.toUInt())\n    /** Multiplies this value by the other value. *^\n    @kotlin.internal.InlineOnly\n
public inline operator fun times(other: UInt): UInt = UInt(this.data.times(other.data))\n    /** Multiplies this value
by the other value. *^\n    @kotlin.internal.InlineOnly\n    public inline operator fun times(other: ULong): ULong =
this.toULong().times(other)\n\n    /** Divides this value by the other value, truncating the result to an integer that is
closer to zero. *^\n    @kotlin.internal.InlineOnly\n    public inline operator fun div(other: UByte): UInt =
this.div(other.toUInt())\n    /** Divides this value by the other value, truncating the result to an integer that is closer
to zero. *^\n    @kotlin.internal.InlineOnly\n    public inline operator fun div(other: UShort): UInt =
this.div(other.toUInt())\n    /** Divides this value by the other value, truncating the result to an integer that is closer
to zero. *^\n    @kotlin.internal.InlineOnly\n    public inline operator fun div(other: UInt): UInt = uintDivide(this,
other)\n    /** Divides this value by the other value, truncating the result to an integer that is closer to zero. *^\n
    @kotlin.internal.InlineOnly\n    public inline operator fun div(other: ULong): ULong =
this.toULong().div(other)\n\n    /**\n    * Calculates the remainder of truncating division of this value by the other
value.\n    * \n    * The result is always less than the divisor.\n    *^\n    @kotlin.internal.InlineOnly\n    public
inline operator fun rem(other: UByte): UInt = this.rem(other.toUInt())\n    /**\n    * Calculates the remainder of
truncating division of this value by the other value.\n    * \n    * The result is always less than the divisor.\n    *^\n

```

```

@kotlin.internal.InlineOnly\n public inline operator fun rem(other: UShort): UInt = this.rem(other.toUInt())\n
/**\n * Calculates the remainder of truncating division of this value by the other value.\n * \n * The result is
always less than the divisor.\n */\n @kotlin.internal.InlineOnly\n public inline operator fun rem(other: UInt):
UInt = uintRemainder(this, other)\n /**\n * Calculates the remainder of truncating division of this value by the
other value.\n * \n * The result is always less than the divisor.\n */\n @kotlin.internal.InlineOnly\n public
inline operator fun rem(other: ULong): ULong = this.toULong().rem(other)\n\n /**\n * Divides this value by
the other value, flooring the result to an integer that is closer to negative infinity.\n * \n * For unsigned types,
the results of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n
public inline fun floorDiv(other: UByte): UInt = this.floorDiv(other.toUInt())\n /**\n * Divides this value by the
other value, flooring the result to an integer that is closer to negative infinity.\n * \n * For unsigned types, the
results of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n public
inline fun floorDiv(other: UShort): UInt = this.floorDiv(other.toUInt())\n /**\n * Divides this value by the other
value, flooring the result to an integer that is closer to negative infinity.\n * \n * For unsigned types, the results
of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n public inline
fun floorDiv(other: UInt): UInt = div(other)\n /**\n * Divides this value by the other value, flooring the result to
an integer that is closer to negative infinity.\n * \n * For unsigned types, the results of flooring division and
truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n public inline fun floorDiv(other:
ULong): ULong = this.toULong().floorDiv(other)\n\n /**\n * Calculates the remainder of flooring division of
this value by the other value.\n * \n * The result is always less than the divisor.\n * \n * For unsigned
types, the remainders of flooring division and truncating division are the same.\n */\n
@kotlin.internal.InlineOnly\n public inline fun mod(other: UByte): UByte = this.mod(other.toUInt()).toUByte()\n
/**\n * Calculates the remainder of flooring division of this value by the other value.\n * \n * The result is
always less than the divisor.\n * \n * For unsigned types, the remainders of flooring division and truncating
division are the same.\n */\n @kotlin.internal.InlineOnly\n public inline fun mod(other: UShort): UShort =
this.mod(other.toUInt()).toUShort()\n /**\n * Calculates the remainder of flooring division of this value by the
other value.\n * \n * The result is always less than the divisor.\n * \n * For unsigned types, the remainders
of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n public inline
fun mod(other: UInt): UInt = rem(other)\n /**\n * Calculates the remainder of flooring division of this value by
the other value.\n * \n * The result is always less than the divisor.\n * \n * For unsigned types, the
remainders of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n
public inline fun mod(other: ULong): ULong = this.toULong().mod(other)\n\n /**\n * Returns this value
incremented by one.\n */\n @sample samples.misc.Builtins.inc\n */\n @kotlin.internal.InlineOnly\n
public inline operator fun inc(): UInt = UInt(data.inc())\n\n /**\n * Returns this value decremented by one.\n
*/\n @sample samples.misc.Builtins.dec\n */\n @kotlin.internal.InlineOnly\n public inline operator fun
dec(): UInt = UInt(data.dec())\n\n /**\n * Creates a range from this value to the specified [other] value. */\n
@kotlin.internal.InlineOnly\n public inline operator fun rangeTo(other: UInt): UIntRange = UIntRange(this,
other)\n\n /**\n * Shifts this value left by the [bitCount] number of bits.\n * \n * Note that only the five
lowest-order bits of the [bitCount] are used as the shift distance.\n * \n * The shift distance actually used is therefore
always in the range `0..31`.\n */\n @kotlin.internal.InlineOnly\n public inline infix fun shl(bitCount: Int): UInt
= UInt(data shl bitCount)\n\n /**\n * Shifts this value right by the [bitCount] number of bits, filling the leftmost
bits with zeros.\n * \n * Note that only the five lowest-order bits of the [bitCount] are used as the shift
distance.\n * \n * The shift distance actually used is therefore always in the range `0..31`.\n */\n
@kotlin.internal.InlineOnly\n public inline infix fun shr(bitCount: Int): UInt = UInt(data ushr bitCount)\n\n /**
Performs a bitwise AND operation between the two values. */\n @kotlin.internal.InlineOnly\n public inline infix
fun and(other: UInt): UInt = UInt(this.data and other.data)\n\n /**\n * Performs a bitwise OR operation between the two
values. */\n @kotlin.internal.InlineOnly\n public inline infix fun or(other: UInt): UInt = UInt(this.data or
other.data)\n\n /**\n * Performs a bitwise XOR operation between the two values. */\n @kotlin.internal.InlineOnly\n
public inline infix fun xor(other: UInt): UInt = UInt(this.data xor other.data)\n\n /**\n * Inverts the bits in this value.

```

```

*  

@kotlin.internal.InlineOnly  

public inline fun inv(): UInt = UInt(data.inv())  

/**  

 * Converts this [UInt] value to [Byte].  

 * If this value is less than or equals to [Byte.MAX_VALUE], the resulting `Byte` value represents  

 * the same numerical value as this `UInt`.  

 * The resulting `Byte` value is represented by the least significant 8 bits of this `UInt` value.  

 * Note that the resulting `Byte` value may be negative.  

 */  

@kotlin.internal.InlineOnly  

public inline fun toByte(): Byte = data.toByte()  

/**  

 * Converts this [UInt] value to [Short].  

 * If this value is less than or equals to [Short.MAX_VALUE], the resulting `Short` value represents  

 * the same numerical value as this `UInt`.  

 * The resulting `Short` value is represented by the least significant 16 bits of this `UInt` value.  

 * Note that the resulting `Short` value may be negative.  

 */  

@kotlin.internal.InlineOnly  

public inline fun toShort(): Short = data.toShort()  

/**  

 * Converts this [UInt] value to [Int].  

 * If this value is less than or equals to [Int.MAX_VALUE], the resulting `Int` value represents  

 * the same numerical value as this `UInt`. Otherwise the result is negative.  

 * The resulting `Int` value has the same binary representation as this `UInt` value.  

 */  

@kotlin.internal.InlineOnly  

public inline fun toInt(): Int = data  

/**  

 * Converts this [UInt] value to [Long].  

 * The resulting `Long` value represents the same numerical value as this `UInt`.  

 * The least significant 32 bits of the resulting `Long` value are the same as the bits of this `UInt` value,  

 * whereas the most significant 32 bits are filled with zeros.  

 */  

@kotlin.internal.InlineOnly  

public inline fun toLong(): Long = data.toLong() and 0xFFFF_FFFF  

/**  

 * Converts this [UInt] value to [UByte].  

 * If this value is less than or equals to [UByte.MAX_VALUE], the resulting `UByte` value represents  

 * the same numerical value as this `UInt`.  

 * The resulting `UByte` value is represented by the least significant 8 bits of this `UInt` value.  

 */  

@kotlin.internal.InlineOnly  

public inline fun toUByte(): UByte = data.toUByte()  

/**  

 * Converts this [UInt] value to [UShort].  

 * If this value is less than or equals to [UShort.MAX_VALUE], the resulting `UShort` value represents  

 * the same numerical value as this `UInt`.  

 * The resulting `UShort` value is represented by the least significant 16 bits of this `UInt` value.  

 */  

@kotlin.internal.InlineOnly  

public inline fun toUShort(): UShort = data.toUShort()  

/** Returns this value. */  

@kotlin.internal.InlineOnly  

public inline fun toUInt(): UInt = this  

/**  

 * Converts this [UInt] value to [ULong].  

 * The resulting `ULong` value represents the same numerical value as this `UInt`.  

 * The least significant 32 bits of the resulting `ULong` value are the same as the bits of this `UInt` value,  

 * whereas the most significant 32 bits are filled with zeros.  

 */  

@kotlin.internal.InlineOnly  

public inline fun toULong(): ULong = ULong(data.toLong() and 0xFFFF_FFFF)  

/**  

 * Converts this [UInt] value to [Float].  

 * The resulting value is the closest `Float` to this `UInt` value.  

 * In case when this `UInt` value is exactly between two `Float`s,  

 * the one with zero at least significant bit of mantissa is selected.  

 */  

@kotlin.internal.InlineOnly  

public inline fun toFloat(): Float = this.toDouble().toFloat()  

/**  

 * Converts this [UInt] value to [Double].  

 * The resulting `Double` value represents the same numerical value as this `UInt`.  

 */  

@kotlin.internal.InlineOnly  

public inline fun toDouble(): Double = UIntToDouble(data)  

public override fun toString(): String = toLong().toString()  

/**  

 * Converts this [Byte] value to [UInt].  

 * If this value is positive, the resulting `UInt` value represents the same numerical value as this `Byte`.  

 * The least significant 8 bits of the resulting `UInt` value are the same as the bits of this `Byte` value,  

 * whereas the most significant 24 bits are filled with the sign bit of this value.  

 */  

@SinceKotlin("1.5")  

@WasExperimental(ExperimentalUnsignedTypes::class)  

@kotlin.internal.InlineOnly  

public inline fun Byte.toUInt(): UInt = UInt(this.toInt())  

/**  

 * Converts this [Short] value to [UInt].  

 * If this value is positive, the resulting `UInt` value represents the same numerical value as this `Short`.  

 * The least significant 16 bits of the resulting `UInt` value are the same as the bits of this `Short` value,  

 * whereas the most significant 16 bits are filled with the sign bit of this value.  

 */  

@SinceKotlin("1.5")  

@WasExperimental(ExperimentalUnsignedTypes::class)  

@kotlin.internal.InlineOnly  

public inline fun Short.toUInt(): UInt = UInt(this.toInt())  

/**  

 * Converts this [Int] value to [UInt].  

 * If this value is positive, the resulting `UInt` value represents the same numerical value as this `Int`.  

 * The resulting `UInt` value has the same binary representation as this `Int` value.  

 */  

@SinceKotlin("1.5")  

@WasExperimental(ExperimentalUnsignedTypes::class)  

@kotlin.internal.InlineOnly

```

```

public inline fun Int.toUInt(): UInt = UInt(this)
    * Converts this [Long] value to [UInt].
    * If this value is positive and less than or equals to [UInt.MAX_VALUE], the resulting `UInt` value represents the same numerical value as this `Long`.
    * The resulting `UInt` value is represented by the least significant 32 bits of this `Long` value.

@SinceKotlin("1.5")@WasExperimental(ExperimentalUnsignedTypes::class)@kotlin.internal.InlineOnly
public inline fun Long.toUInt(): UInt = UInt(this.toInt())
    * Converts this [Float] value to [UInt].
    * The fractional part, if any, is rounded down towards zero.
    * Returns zero if this `Float` value is negative or `NaN`, [UInt.MAX_VALUE] if it's bigger than `UInt.MAX_VALUE`.

@SinceKotlin("1.5")@WasExperimental(ExperimentalUnsignedTypes::class)@kotlin.internal.InlineOnly
public inline fun Float.toUInt(): UInt = doubleToUInt(this.toDouble())
    * Converts this [Double] value to [UInt].
    * The fractional part, if any, is rounded down towards zero.
    * Returns zero if this `Double` value is negative or `NaN`, [UInt.MAX_VALUE] if it's bigger than `UInt.MAX_VALUE`.

@SinceKotlin("1.5")@WasExperimental(ExperimentalUnsignedTypes::class)@kotlin.internal.InlineOnly
public inline fun Double.toUInt(): UInt = doubleToUInt(this)
    * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.
    * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.
    * Auto-generated file. DO NOT EDIT!

package kotlin

import kotlin.experimental.*

kotlin.jvm.*
@SinceKotlin("1.5")@WasExperimental(ExperimentalUnsignedTypes::class)@JvmInline
public value class UShort @PublishedApi internal constructor(@PublishedApi internal val data: Short) : Comparable<UShort> {
    companion object {
        /**
         * A constant holding the minimum value an instance of UShort can have.
         */
        public const val MIN_VALUE: UShort = UShort(0)
        /**
         * A constant holding the maximum value an instance of UShort can have.
         */
        public const val MAX_VALUE: UShort = UShort(-1)
        /**
         * The number of bytes used to represent an instance of UShort in a binary form.
         */
        public const val SIZE_BYTES: Int = 2
        /**
         * The number of bits used to represent an instance of UShort in a binary form.
         */
        public const val SIZE_BITS: Int = 16
    }
    /**
     * Compares this value with the specified value for order.
     * Returns zero if this value is equal to the specified other value, a negative number if it's less than other,
     * or a positive number if it's greater than other.
     */
    @kotlin.internal.InlineOnly
    public inline operator fun compareTo(other: UByte): Int = this.toInt().compareTo(other.toInt())
    /**
     * Compares this value with the specified value for order.
     * Returns zero if this value is equal to the specified other value, a negative number if it's less than other,
     * or a positive number if it's greater than other.
     */
    @kotlin.internal.InlineOnly
    public inline operator fun compareTo(other: UShort): Int = this.toInt().compareTo(other.toInt())
    /**
     * Compares this value with the specified value for order.
     * Returns zero if this value is equal to the specified other value, a negative number if it's less than other,
     * or a positive number if it's greater than other.
     */
    @kotlin.internal.InlineOnly
    public inline operator fun compareTo(other: UInt): Int = this.toUInt().compareTo(other)
    /**
     * Compares this value with the specified value for order.
     * Returns zero if this value is equal to the specified other value, a negative number if it's less than other,
     * or a positive number if it's greater than other.
     */
    @kotlin.internal.InlineOnly
    public inline operator fun compareTo(other: ULong): Int = this.toULong().compareTo(other)
    /**
     * Adds the other value to this value.
     */
    @kotlin.internal.InlineOnly
    public inline operator fun plus(other: UByte): UInt = this.toUInt().plus(other.toUInt())
    /**
     * Adds the other value to this value.
     */
    @kotlin.internal.InlineOnly
    public inline operator fun plus(other: UShort): UInt = this.toUInt().plus(other.toUInt())
    /**
     * Adds the other value to this value.
     */
    @kotlin.internal.InlineOnly
    public inline operator fun plus(other: UInt): UInt = this.toUInt().plus(other)
    /**
     * Adds the other value to this value.
     */
    @kotlin.internal.InlineOnly
    public inline operator fun plus(other: ULong): ULong = this.toULong().plus(other)
    /**
     * Subtracts the other value from this value.
     */
    @kotlin.internal.InlineOnly
    public inline operator fun minus(other: UByte): UInt = this.toUInt().minus(other.toUInt())
    /**
     * Subtracts the other value from this value.
     */
    @kotlin.internal.InlineOnly
    public inline operator fun minus(other: UShort): UInt =

```

```

this.toUInt().minus(other.toUInt())\n /** Subtracts the other value from this value. */\n
@kotlin.internal.InlineOnly\n public inline operator fun minus(other: UInt): UInt = this.toUInt().minus(other)\n
/** Subtracts the other value from this value. */\n @kotlin.internal.InlineOnly\n public inline operator fun
minus(other: ULong): ULong = this.toULong().minus(other)\n\n /** Multiplies this value by the other value. */\n
@kotlin.internal.InlineOnly\n public inline operator fun times(other: UByte): UInt =
this.toUInt().times(other.toUInt())\n /** Multiplies this value by the other value. */\n
@kotlin.internal.InlineOnly\n public inline operator fun times(other: UShort): UInt =
this.toUInt().times(other.toUInt())\n /** Multiplies this value by the other value. */\n
@kotlin.internal.InlineOnly\n public inline operator fun times(other: UInt): UInt = this.toUInt().times(other)\n
/** Multiplies this value by the other value. */\n @kotlin.internal.InlineOnly\n public inline operator fun
times(other: ULong): ULong = this.toULong().times(other)\n\n /** Divides this value by the other value,
truncating the result to an integer that is closer to zero. */\n @kotlin.internal.InlineOnly\n public inline operator
fun div(other: UByte): UInt = this.toUInt().div(other.toUInt())\n /** Divides this value by the other value,
truncating the result to an integer that is closer to zero. */\n @kotlin.internal.InlineOnly\n public inline operator
fun div(other: UShort): UInt = this.toUInt().div(other.toUInt())\n /** Divides this value by the other value,
truncating the result to an integer that is closer to zero. */\n @kotlin.internal.InlineOnly\n public inline operator
fun div(other: UInt): UInt = this.toUInt().div(other)\n /** Divides this value by the other value, truncating the
result to an integer that is closer to zero. */\n @kotlin.internal.InlineOnly\n public inline operator fun div(other:
ULong): ULong = this.toULong().div(other)\n\n /**\n * Calculates the remainder of truncating division of this
value by the other value.\n * \n * The result is always less than the divisor.\n */\n
@kotlin.internal.InlineOnly\n public inline operator fun rem(other: UByte): UInt =
this.toUInt().rem(other.toUInt())\n /**\n * Calculates the remainder of truncating division of this value by the
other value.\n * \n * The result is always less than the divisor.\n */\n @kotlin.internal.InlineOnly\n public
inline operator fun rem(other: UShort): UInt = this.toUInt().rem(other.toUInt())\n /**\n * Calculates the
remainder of truncating division of this value by the other value.\n * \n * The result is always less than the
divisor.\n */\n @kotlin.internal.InlineOnly\n public inline operator fun rem(other: UInt): UInt =
this.toUInt().rem(other)\n /**\n * Calculates the remainder of truncating division of this value by the other
value.\n * \n * The result is always less than the divisor.\n */\n @kotlin.internal.InlineOnly\n public
inline operator fun rem(other: ULong): ULong = this.toULong().rem(other)\n\n /**\n * Divides this value by
the other value, flooring the result to an integer that is closer to negative infinity.\n * \n * For unsigned types,
the results of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n
public inline fun floorDiv(other: UByte): UInt = this.toUInt().floorDiv(other.toUInt())\n /**\n * Divides this
value by the other value, flooring the result to an integer that is closer to negative infinity.\n * \n * For unsigned
types, the results of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n
public inline fun floorDiv(other: UShort): UInt = this.toUInt().floorDiv(other.toUInt())\n /**\n * Divides this
value by the other value, flooring the result to an integer that is closer to negative infinity.\n * \n * For unsigned
types, the results of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n
public inline fun floorDiv(other: UInt): UInt = this.toUInt().floorDiv(other)\n /**\n * Divides this value by the
other value, flooring the result to an integer that is closer to negative infinity.\n * \n * For unsigned types, the
results of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n public
inline fun floorDiv(other: ULong): ULong = this.toULong().floorDiv(other)\n\n /**\n * Calculates the
remainder of flooring division of this value by the other value.\n * \n * The result is always less than the
divisor.\n * \n * For unsigned types, the remainders of flooring division and truncating division are the same.\n
*/\n @kotlin.internal.InlineOnly\n public inline fun mod(other: UByte): UByte =
this.toUInt().mod(other.toUInt()).toUByte()\n /**\n * Calculates the remainder of flooring division of this value
by the other value.\n * \n * The result is always less than the divisor.\n * \n * For unsigned types, the
remainders of flooring division and truncating division are the same.\n */\n @kotlin.internal.InlineOnly\n
public inline fun mod(other: UShort): UShort = this.toUInt().mod(other.toUInt()).toUShort()\n /**\n *

```

Calculates the remainder of flooring division of this value by the other value. `\n * \n * The result is always less than the divisor.\n * \n * For unsigned types, the remainders of flooring division and truncating division are the same.\n * \n @kotlin.internal.InlineOnly\n public inline fun mod(other: UInt): UInt = this.toUInt().mod(other)\n /**\n * Calculates the remainder of flooring division of this value by the other value.\n * \n * The result is always less than the divisor.\n * \n * For unsigned types, the remainders of flooring division and truncating division are the same.\n * \n @kotlin.internal.InlineOnly\n public inline fun mod(other: ULong): ULong = this.toULong().mod(other)\n /**\n * Returns this value incremented by one.\n * \n * @sample samples.misc.Builtins.inc\n * \n @kotlin.internal.InlineOnly\n public inline operator fun inc(): UShort = UShort(data.inc())\n /**\n * Returns this value decremented by one.\n * \n * @sample samples.misc.Builtins.dec\n * \n @kotlin.internal.InlineOnly\n public inline operator fun dec(): UShort = UShort(data.dec())\n /**\n * Creates a range from this value to the specified [other] value. *\n @kotlin.internal.InlineOnly\n public inline operator fun rangeTo(other: UShort): UIntRange = UIntRange(this.toUInt(), other.toUInt())\n /**\n * Performs a bitwise AND operation between the two values. *\n @kotlin.internal.InlineOnly\n public inline infix fun and(other: UShort): UShort = UShort(this.data and other.data)\n /**\n * Performs a bitwise OR operation between the two values. *\n @kotlin.internal.InlineOnly\n public inline infix fun or(other: UShort): UShort = UShort(this.data or other.data)\n /**\n * Performs a bitwise XOR operation between the two values. *\n @kotlin.internal.InlineOnly\n public inline infix fun xor(other: UShort): UShort = UShort(this.data xor other.data)\n /**\n * Inverts the bits in this value. *\n @kotlin.internal.InlineOnly\n public inline fun inv(): UShort = UShort(data.inv())\n /**\n * Converts this [UShort] value to [Byte].\n * \n * If this value is less than or equals to [Byte.MAX_VALUE], the resulting `Byte` value represents\n * the same numerical value as this `UShort`.\n * \n * The resulting `Byte` value is represented by the least significant 8 bits of this `UShort` value.\n * \n * Note that the resulting `Byte` value may be negative.\n * \n @kotlin.internal.InlineOnly\n public inline fun toByte(): Byte = data.toByte()\n /**\n * Converts this [UShort] value to [Short].\n * \n * If this value is less than or equals to [Short.MAX_VALUE], the resulting `Short` value represents\n * the same numerical value as this `UShort`. Otherwise the result is negative.\n * \n * The resulting `Short` value has the same binary representation as this `UShort` value.\n * \n @kotlin.internal.InlineOnly\n public inline fun toShort(): Short = data\n /**\n * Converts this [UShort] value to [Int].\n * \n * The resulting `Int` value represents the same numerical value as this `UShort`.\n * \n * The least significant 16 bits of the resulting `Int` value are the same as the bits of this `UShort` value,\n * whereas the most significant 16 bits are filled with zeros.\n * \n @kotlin.internal.InlineOnly\n public inline fun toInt(): Int = data.toInt() and 0xFFFF\n /**\n * Converts this [UShort] value to [Long].\n * \n * The resulting `Long` value represents the same numerical value as this `UShort`.\n * \n * The least significant 16 bits of the resulting `Long` value are the same as the bits of this `UShort` value,\n * whereas the most significant 48 bits are filled with zeros.\n * \n @kotlin.internal.InlineOnly\n public inline fun toLong(): Long = data.toLong() and 0xFFFF\n /**\n * Converts this [UShort] value to [UByte].\n * \n * If this value is less than or equals to [UByte.MAX_VALUE], the resulting `UByte` value represents\n * the same numerical value as this `UShort`.\n * \n * The resulting `UByte` value is represented by the least significant 8 bits of this `UShort` value.\n * \n @kotlin.internal.InlineOnly\n public inline fun toUByte(): UByte = data.toUByte()\n /**\n * Returns this value. *\n @kotlin.internal.InlineOnly\n public inline fun toUShort(): UShort = this\n /**\n * Converts this [UShort] value to [UInt].\n * \n * The resulting `UInt` value represents the same numerical value as this `UShort`.\n * \n * The least significant 16 bits of the resulting `UInt` value are the same as the bits of this `UShort` value,\n * whereas the most significant 16 bits are filled with zeros.\n * \n @kotlin.internal.InlineOnly\n public inline fun toUInt(): UInt = UInt(data.toInt() and 0xFFFF)\n /**\n * Converts this [UShort] value to [ULong].\n * \n * The resulting `ULong` value represents the same numerical value as this `UShort`.\n * \n * The least significant 16 bits of the resulting `ULong` value are the same as the bits of this `UShort` value,\n * whereas the most significant 48 bits are filled with zeros.\n * \n @kotlin.internal.InlineOnly\n public inline fun toULong(): ULong = ULong(data.toLong() and 0xFFFF)\n /**\n * Converts this [UShort] value to [Float].\n * \n * The resulting `Float` value represents the same numerical value as this `UShort`.\n * \n`

```

@kotlin.internal.InlineOnly\n public inline fun toFloat(): Float = this.toInt().toFloat()\n /**\n * Converts this
[UShort] value to [Double].\n *\n * The resulting `Double` value represents the same numerical value as this
`UShort`.\n */\n @kotlin.internal.InlineOnly\n public inline fun toDouble(): Double =
this.toInt().toDouble()\n\n public override fun toString(): String = toInt().toString()\n}\n\n/**\n * Converts this
[Byte] value to [UShort].\n *\n * If this value is positive, the resulting `UShort` value represents the same numerical
value as this `Byte`.\n *\n * The least significant 8 bits of the resulting `UShort` value are the same as the bits of this
`Byte` value,\n * whereas the most significant 8 bits are filled with the sign bit of this value.\n
*/\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\n
public inline fun Byte.toUShort(): UShort = UShort(this.toShort())\n/**\n * Converts this [Short] value to
[UShort].\n *\n * If this value is positive, the resulting `UShort` value represents the same numerical value as this
`Short`.\n *\n * The resulting `UShort` value has the same binary representation as this `Short` value.\n
*/\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\n
public inline fun Short.toUShort(): UShort = UShort(this)\n/**\n * Converts this [Int] value to [UShort].\n *\n * If
this value is positive and less than or equals to [UShort.MAX_VALUE], the resulting `UShort` value represents\n *
the same numerical value as this `Int`.\n *\n * The resulting `UShort` value is represented by the least significant 16
bits of this `Int` value.\n
*/\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\n
public inline fun Int.toUShort(): UShort = UShort(this.toShort())\n/**\n * Converts this [Long] value to
[UShort].\n *\n * If this value is positive and less than or equals to [UShort.MAX_VALUE], the resulting `UShort`
value represents\n * the same numerical value as this `Long`.\n *\n * The resulting `UShort` value is represented by
the least significant 16 bits of this `Long` value.\n
*/\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\n
public inline fun Long.toUShort(): UShort = UShort(this.toShort())\n", "/*\n * Copyright 2010-2021 JetBrains s.r.o.
and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license
that can be found in the license/LICENSE.txt file.\n */\n\n// Auto-generated file. DO NOT EDIT!\n\npackage
kotlin.ranges\n\n/**\n * A range of values of type `Char`.\n */\npublic class CharRange(start: Char, endInclusive:
Char) : CharProgression(start, endInclusive, 1), ClosedRange<Char> {\n    override val start: Char get() = first\n
    override val endInclusive: Char get() = last\n\n    override fun contains(value: Char): Boolean = first <= value &&
value <= last\n\n    /**\n     * Checks whether the range is empty.\n     *\n     * The range is empty if its start value is
greater than the end value.\n     */\n    override fun isEmpty(): Boolean = first > last\n\n    override fun equals(other:
Any?): Boolean =\n        other is CharRange && (isEmpty() && other.isEmpty()) ||\n        first == other.first && last
== other.last)\n\n    override fun hashCode(): Int =\n        if (isEmpty()) -1 else (31 * first.code + last.code)\n\n    override fun toString(): String = "$first..$last"\n\n    companion object {\n        /** An empty range of values of
type Char. */\n        public val EMPTY: CharRange = CharRange(1.toChar(), 0.toChar())\n    }\n}\n\n/**\n * A
range of values of type `Int`.\n */\npublic class IntRange(start: Int, endInclusive: Int) : IntProgression(start,
endInclusive, 1), ClosedRange<Int> {\n    override val start: Int get() = first\n    override val endInclusive: Int get() =
last\n\n    override fun contains(value: Int): Boolean = first <= value && value <= last\n\n    /**\n     * Checks
whether the range is empty.\n     *\n     * The range is empty if its start value is greater than the end value.\n     */\n
    override fun isEmpty(): Boolean = first > last\n\n    override fun equals(other: Any?): Boolean =\n        other is
IntRange && (isEmpty() && other.isEmpty()) ||\n        first == other.first && last == other.last)\n\n    override fun
hashCode(): Int =\n        if (isEmpty()) -1 else (31 * first + last)\n\n    override fun toString(): String =
"$first..$last"\n\n    companion object {\n        /** An empty range of values of type Int. */\n        public val
EMPTY: IntRange = IntRange(1, 0)\n    }\n}\n\n/**\n * A range of values of type `Long`.\n */\npublic class
LongRange(start: Long, endInclusive: Long) : LongProgression(start, endInclusive, 1), ClosedRange<Long> {\n
    override val start: Long get() = first\n    override val endInclusive: Long get() = last\n\n    override fun
contains(value: Long): Boolean = first <= value && value <= last\n\n    /**\n     * Checks whether the range is
empty.\n     *\n     * The range is empty if its start value is greater than the end value.\n     */\n    override fun
isEmpty(): Boolean = first > last\n\n    override fun equals(other: Any?): Boolean =\n        other is LongRange &&

```

```

(isEmpty() && other.isEmpty() ||\n    first == other.first && last == other.last)\n\n    override fun hashCode(): Int
= \n    if (isEmpty()) -1 else (31 * (first xor (first ushr 32)) + (last xor (last ushr 32))).toInt()\n\n    override fun
toString(): String = \"$first..$last\"\n\n    companion object {\n        /** An empty range of values of type Long. */\n
        public val EMPTY: LongRange = LongRange(1, 0)\n    }\n\n    /**\n     * Copyright 2010-2021 JetBrains s.r.o.
and Kotlin Programming Language contributors.\n     * Use of this source code is governed by the Apache 2.0 license
that can be found in the license/LICENSE.txt file.\n
*/\n\n    @file:kotlin.jvm.JvmMultifileClass\n    @file:kotlin.jvm.JvmName("CollectionsKt")\n    @file:OptIn(kotlin.exper
imental.ExperimentalTypeInference::class)\n    @package kotlin.collections\n    @import kotlin.contracts.*\n    @import
kotlin.random.Random\n\n    internal object EmptyIterator : ListIterator<Nothing> {\n        override fun hasNext():
Boolean = false\n        override fun hasPrevious(): Boolean = false\n        override fun nextIndex(): Int = 0\n        override
fun previousIndex(): Int = -1\n        override fun next(): Nothing = throw NoSuchElementException()\n        override fun
previous(): Nothing = throw NoSuchElementException()\n    }\n\n    internal object EmptyList : List<Nothing>,
Serializable, RandomAccess {\n        private const val serialVersionUID: Long = -7390468764508069838L\n\n        override fun equals(other: Any?): Boolean = other is List<*> && other.isEmpty()\n        override fun hashCode(): Int
= 1\n        override fun toString(): String = "[]"\n        override val size: Int get() = 0\n        override fun isEmpty():
Boolean = true\n        override fun contains(element: Nothing): Boolean = false\n        override fun containsAll(elements:
Collection<Nothing>): Boolean = elements.isEmpty()\n        override fun get(index: Int): Nothing = throw
IndexOutOfBoundsException("Empty list doesn't contain element at index $index.")\n        override fun
indexOf(element: Nothing): Int = -1\n        override fun lastIndexOf(element: Nothing): Int = -1\n        override fun
iterator(): Iterator<Nothing> = EmptyIterator\n        override fun listIterator(): ListIterator<Nothing> = EmptyIterator\n
        override fun listIterator(index: Int): ListIterator<Nothing> {\n            if (index != 0) throw
IndexOutOfBoundsException("Index: $index")\n            return EmptyIterator\n        }\n        override fun
subList(fromIndex: Int, toIndex: Int): List<Nothing> {\n            if (fromIndex == 0 && toIndex == 0) return this\n
            throw IndexOutOfBoundsException("fromIndex: $fromIndex, toIndex: $toIndex")\n        }\n        private fun
readResolve(): Any = EmptyList\n\n        internal fun <T> Array<out T>.asCollection(): Collection<T> =
ArrayAsCollection(this, isVarargs = false)\n        private class ArrayAsCollection<T>(val values: Array<out T>, val
isVarargs: Boolean) : Collection<T> {\n            override val size: Int get() = values.size\n            override fun isEmpty():
Boolean = values.isEmpty()\n            override fun contains(element: T): Boolean = values.contains(element)\n            override
fun containsAll(elements: Collection<T>): Boolean = elements.all { contains(it) }\n            override fun iterator():
Iterator<T> = values.iterator()\n            // override hidden toArray implementation to prevent copying of values array\n
            public fun toArray(): Array<out Any?> = values.copyToArrayOfAny(isVarargs)\n        }\n\n        /**\n         * Returns an empty
read-only list. The returned list is serializable (JVM).\n         * @sample
samples.collections.Collections.Lists.emptyReadOnlyList\n         */\n        public fun <T> emptyList(): List<T> =
EmptyList\n\n        /**\n         * Returns a new read-only list of given elements. The returned list is serializable (JVM).\n         *
@sample samples.collections.Collections.Lists.readOnlyList\n         */\n        public fun <T> listOf(vararg elements: T):
List<T> = if (elements.size > 0) elements.asList() else emptyList()\n\n        /**\n         * Returns an empty read-only list. The
returned list is serializable (JVM).\n         * @sample samples.collections.Collections.Lists.emptyReadOnlyList\n         */\n        @kotlin.internal.InlineOnly\n        public inline fun <T> listOf(): List<T> = emptyList()\n\n        /**\n         * Returns an empty
new [MutableList].\n         * @sample samples.collections.Collections.Lists.emptyMutableList\n         */\n        @SinceKotlin("1.1")\n        @kotlin.internal.InlineOnly\n        public inline fun <T> mutableListOf(): MutableList<T> =
ArrayList()\n\n        /**\n         * Returns an empty new [ArrayList].\n         * @sample
samples.collections.Collections.Lists.emptyArrayList\n         */\n        @SinceKotlin("1.1")\n        @kotlin.internal.InlineOnly\n        public inline fun <T> arrayListOf(): ArrayList<T> =
ArrayList()\n\n        /**\n         * Returns a new [MutableList] with the given elements.\n         * @sample
samples.collections.Collections.Lists.mutableList\n         */\n        public fun <T> mutableListOf(vararg elements: T):
MutableList<T> =\n            if (elements.size == 0) ArrayList() else ArrayList(ArrayAsCollection(elements, isVarargs =
true))\n\n        /**\n         * Returns a new [ArrayList] with the given elements.\n         * @sample
samples.collections.Collections.Lists.arrayList\n         */\n        public fun <T> arrayListOf(vararg elements: T): ArrayList<T>

```

```

= \n if (elements.size == 0) ArrayList() else ArrayList(ArrayAsCollection(elements, isVarargs = true)) \n \n \n *
Returns a new read-only list either of single given element, if it is not null, or empty list if the element is null. The
returned list is serializable (JVM). \n * @sample samples.collections.Collections.Lists.listOfNotNull \n * \n public fun
<T : Any> listOfNotNull(element: T?): List<T> = if (element != null) listOf(element) else emptyList() \n \n \n *
Returns a new read-only list only of those given elements, that are not null. The returned list is serializable
(JVM). \n * @sample samples.collections.Collections.Lists.listOfNotNull \n * \n public fun <T : Any>
listOfNotNull(vararg elements: T?): List<T> = elements.filterNotNull() \n \n \n * Creates a new read-only list with
the specified [size], where each element is calculated by calling the specified \n * [init] function. \n * \n * The
function [init] is called for each list element sequentially starting from the first one. \n * It should return the value for
a list element given its index. \n * \n * @sample samples.collections.Collections.Lists.readOnlyListFromInitializer \n
* \n @SinceKotlin("1.1") \n @kotlin.internal.InlineOnly \n public inline fun <T> List(size: Int, init: (index: Int) -> T):
List<T> = MutableList(size, init) \n \n \n * Creates a new mutable list with the specified [size], where each element
is calculated by calling the specified \n * [init] function. \n * \n * The function [init] is called for each list element
sequentially starting from the first one. \n * It should return the value for a list element given its index. \n * \n *
@sample samples.collections.Collections.Lists.mutableListFromInitializer \n
* \n @SinceKotlin("1.1") \n @kotlin.internal.InlineOnly \n public inline fun <T> MutableList(size: Int, init: (index:
Int) -> T): MutableList<T> { \n val list = ArrayList<T>(size) \n repeat(size) { index -> list.add(init(index)) } \n
return list \n } \n \n \n * Builds a new read-only [List] by populating a [MutableList] using the given
[builderAction] \n * and returning a read-only list with the same elements. \n * \n * The list passed as a receiver to the
[builderAction] is valid only inside that function. \n * Using it outside of the function produces an unspecified
behavior. \n * \n * The returned list is serializable (JVM). \n * \n * @sample
samples.collections.Builders.Lists.buildListSample \n
* \n @SinceKotlin("1.6") \n @WasExperimental(ExperimentalStdlibApi::class) \n @kotlin.internal.InlineOnly \n publi
c inline fun <E> buildList(@BuilderInference builderAction: MutableList<E>.() -> Unit): List<E> { \n contract {
callsInPlace(builderAction, InvocationKind.EXACTLY_ONCE) } \n return
buildListInternal(builderAction) \n } \n \n @PublishedApi \n @SinceKotlin("1.3") \n @kotlin.internal.InlineOnly \n ninter
nal expect inline fun <E> buildListInternal(builderAction: MutableList<E>.() -> Unit): List<E> \n \n \n * Builds a
new read-only [List] by populating a [MutableList] using the given [builderAction] \n * and returning a read-only list
with the same elements. \n * \n * The list passed as a receiver to the [builderAction] is valid only inside that
function. \n * Using it outside of the function produces an unspecified behavior. \n * \n * The returned list is
serializable (JVM). \n * \n * [capacity] is used to hint the expected number of elements added in the
[builderAction]. \n * \n * @throws IllegalArgumentException if the given [capacity] is negative. \n * \n * @sample
samples.collections.Builders.Lists.buildListSampleWithCapacity \n
* \n @SinceKotlin("1.6") \n @WasExperimental(ExperimentalStdlibApi::class) \n @kotlin.internal.InlineOnly \n publi
c inline fun <E> buildList(capacity: Int, @BuilderInference builderAction: MutableList<E>.() -> Unit): List<E> { \n
contract { callsInPlace(builderAction, InvocationKind.EXACTLY_ONCE) } \n return buildListInternal(capacity,
builderAction) \n } \n \n @PublishedApi \n @SinceKotlin("1.3") \n @kotlin.internal.InlineOnly \n ninternal expect inline
fun <E> buildListInternal(capacity: Int, builderAction: MutableList<E>.() -> Unit): List<E> \n \n \n * Returns an
[IntRange] of the valid indices for this collection. \n * @sample
samples.collections.Collections.Collections.indicesOfCollection \n * \n public val Collection<*>.indices: IntRange \n
get() = 0..size - 1 \n \n \n * Returns the index of the last item in the list or -1 if the list is empty. \n * \n * @sample
samples.collections.Collections.Lists.lastIndexOfList \n * \n public val <T> List<T>.lastIndex: Int \n get() =
this.size - 1 \n \n \n * Returns `true` if the collection is not empty. \n * @sample
samples.collections.Collections.Collections.collectionIsNotEmpty \n * \n @kotlin.internal.InlineOnly \n public inline
fun <T> Collection<T>.isNotEmpty(): Boolean = !isEmpty() \n \n \n * Returns `true` if this nullable collection is
either null or empty. \n * @sample samples.collections.Collections.Collections.collectionIsNullOrEmpty \n
* \n @SinceKotlin("1.3") \n @kotlin.internal.InlineOnly \n public inline fun <T> Collection<T>?.isNullOrEmpty():
Boolean { \n contract { \n returns(false) implies (this@isNullOrEmpty != null) \n } \n return this == null \n }

```

```

this.isEmpty()\n\n**\n * Returns this Collection if it's not `null` and the empty list otherwise.\n * @sample
samples.collections.Collections.Collections.collectionOrEmpty\n */\n@kotlin.internal.InlineOnly\npublic inline fun
<T> Collection<T>?.orEmpty(): Collection<T> = this ?: emptyList()\n\n**\n * Returns this List if it's not `null` and
the empty list otherwise.\n * @sample samples.collections.Collections.Lists.listOrEmpty\n
*/\n@kotlin.internal.InlineOnly\npublic inline fun <T> List<T>?.orEmpty(): List<T> = this ?: emptyList()\n\n**\n
* Returns this collection if it's not empty\n * or the result of calling [defaultValue] function if the collection is
empty.\n *\n * @sample samples.collections.Collections.Collections.collectionIfEmpty\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline fun <C, R> C.ifEmpty(defaultValue: () ->
R): R where C : Collection<*>, C : R =\n    if (isEmpty()) defaultValue() else this\n\n**\n * Checks if all
elements in the specified collection are contained in this collection.\n *\n * Allows to overcome type-safety
restriction of `containsAll` that requires to pass a collection of type `Collection<E>`.\n * @sample
samples.collections.Collections.Collections.collectionContainsAll\n
*/\n@Suppress("EXTENSION_SHADOWED_BY_MEMBER") // false warning, extension takes precedence in
some cases\n@kotlin.internal.InlineOnly\npublic inline fun <@kotlin.internal.OnlyInputTypes T>
Collection<T>.containsAll(elements: Collection<T>): Boolean = this.containsAll(elements)\n\n**\n * Returns a
new list with the elements of this list randomly shuffled\n * using the specified [random] instance as the source of
randomness.\n */\n@SinceKotlin("1.3")\npublic fun <T> Iterable<T>.shuffled(random: Random): List<T> =
toMutableList().apply { shuffle(random) }\n\n\ninternal fun <T> List<T>.optimizeReadOnlyList() = when (size) {\n
    0 -> emptyList()\n    1 -> listOf(this[0])\n    else -> this\n}\n\n**\n * Searches this list or its range for the provided
[element] using the binary search algorithm.\n * The list is expected to be sorted into ascending order according to
the Comparable natural ordering of its elements,\n * otherwise the result is undefined.\n *\n * If the list contains
multiple elements equal to the specified [element], there is no guarantee which one will be found.\n *\n * `null`
value is considered to be less than any non-null value.\n *\n * @return the index of the element, if it is contained in
the list within the specified range;\n * otherwise, the inverted insertion point `(-insertion point - 1)`.\n * The
insertion point is defined as the index at which the element should be inserted,\n * so that the list (or the specified
subrange of list) still remains sorted.\n * @sample
samples.collections.Collections.Lists.binarySearchOnComparable\n * @sample
samples.collections.Collections.Lists.binarySearchWithBoundaries\n */\npublic fun <T : Comparable<T>>
List<T?>.binarySearch(element: T?, fromIndex: Int = 0, toIndex: Int = size): Int {\n    rangeCheck(size, fromIndex,
toIndex)\n    var low = fromIndex\n    var high = toIndex - 1\n    while (low <= high) {\n        val mid = (low +
high).ushr(1) // safe from overflows\n        val midVal = get(mid)\n        val cmp = compareValues(midVal,
element)\n        if (cmp < 0)\n            low = mid + 1\n        else if (cmp > 0)\n            high = mid - 1\n        else\n
            return mid // key found\n    }\n    return -(low + 1) // key not found\n}\n\n**\n * Searches this list or its range
for the provided [element] using the binary search algorithm.\n * The list is expected to be sorted into ascending
order according to the specified [comparator],\n * otherwise the result is undefined.\n *\n * If the list contains
multiple elements equal to the specified [element], there is no guarantee which one will be found.\n *\n * `null`
value is considered to be less than any non-null value.\n *\n * @return the index of the element, if it is contained in
the list within the specified range;\n * otherwise, the inverted insertion point `(-insertion point - 1)`.\n * The
insertion point is defined as the index at which the element should be inserted,\n * so that the list (or the specified
subrange of list) still remains sorted according to the specified [comparator].\n * @sample
samples.collections.Collections.Lists.binarySearchWithComparator\n */\npublic fun <T>
List<T>.binarySearch(element: T, comparator: Comparator<in T>, fromIndex: Int = 0, toIndex: Int = size): Int {\n
    rangeCheck(size, fromIndex, toIndex)\n    var low = fromIndex\n    var high = toIndex - 1\n    while (low <=
high) {\n        val mid = (low + high).ushr(1) // safe from overflows\n        val midVal = get(mid)\n        val cmp =
comparator.compare(midVal, element)\n        if (cmp < 0)\n            low = mid + 1\n        else if (cmp > 0)\n
            high = mid - 1\n        else\n            return mid // key found\n    }\n    return -(low + 1) // key not found\n}\n\n**\n
* Searches this list or its range for an element having the key returned by the specified [selector] function\n * equal to
the provided [key] value using the binary search algorithm.\n * The list is expected to be sorted into ascending order

```

according to the Comparable natural ordering of keys of its elements. \n \* otherwise the result is undefined. \n \* \n \* If the list contains multiple elements with the specified [key], there is no guarantee which one will be found. \n \* \n \* `null` value is considered to be less than any non-null value. \n \* \n \* @return the index of the element with the specified [key], if it is contained in the list within the specified range; \n \* otherwise, the inverted insertion point `(-insertion point - 1)`. \n \* The insertion point is defined as the index at which the element should be inserted, \n \* so that the list (or the specified subrange of list) still remains sorted. \n \* @sample

```

samples.collections.Collections.Lists.binarySearchByKey\n * \n public inline fun <T, K : Comparable<K>>
List<T>.binarySearchBy(\n key: K?,\n fromIndex: Int = 0,\n toIndex: Int = size,\n crossinline selector: (T) ->
K?)\n): Int =\n    binarySearch(fromIndex, toIndex) { compareValues(selector(it), key) }\n\n// do not introduce this
overload --- too rare\n//public fun <T, K> List<T>.binarySearchBy(key: K, comparator: Comparator<K>,
fromIndex: Int = 0, toIndex: Int = size(), selector: (T) -> K): Int =\n//    binarySearch(fromIndex, toIndex) {
comparator.compare(selector(it), key) }\n\n\n/**\n * Searches this list or its range for an element for which the
given [comparison] function returns zero using the binary search algorithm.\n * The list is expected to be sorted
so that the signs of the [comparison] function's return values ascend on the list elements,\n * i.e. negative values
come before zero and zeroes come before positive values.\n * Otherwise, the result is undefined.\n * If the list
contains multiple elements for which [comparison] returns zero, there is no guarantee which one will be found.\n * \n
* @param comparison function that returns zero when called on the list element being searched.\n * On the
elements coming before the target element, the function must return negative values;\n * on the elements coming
after the target element, the function must return positive values.\n * \n * @return the index of the found element, if
it is contained in the list within the specified range;\n * otherwise, the inverted insertion point `(-insertion point -
1)`. \n * The insertion point is defined as the index at which the element should be inserted,\n * so that the list (or the
specified subrange of list) still remains sorted.\n * @sample
samples.collections.Collections.Lists.binarySearchWithComparisonFunction\n * \n public fun <T>
List<T>.binarySearch(fromIndex: Int = 0, toIndex: Int = size, comparison: (T) -> Int): Int {\n    rangeCheck(size,
fromIndex, toIndex)\n\n    var low = fromIndex\n    var high = toIndex - 1\n\n    while (low <= high) {\n        val mid
= (low + high).ushr(1) // safe from overflows\n        val midVal = get(mid)\n        val cmp = comparison(midVal)\n\n        if (cmp < 0)\n            low = mid + 1\n        else if (cmp > 0)\n            high = mid - 1\n        else\n            return
mid // key found\n    }\n    return -(low + 1) // key not found\n}\n\n\n/**\n * Checks that `from` and `to` are in\n * the
range of [0..size] and throws an appropriate exception, if they aren't.\n * \n private fun rangeCheck(size: Int,
fromIndex: Int, toIndex: Int) {\n    when {\n        fromIndex > toIndex -> throw
IllegalArgumentException("\nfromIndex ($fromIndex) is greater than toIndex ($toIndex).")\n        fromIndex < 0 ->
throw IndexOutOfBoundsException("\nfromIndex ($fromIndex) is less than zero.")\n        toIndex > size -> throw
IndexOutOfBoundsException("\ntoIndex ($toIndex) is greater than size ($size).")\n    }\n}\n\n\n@PublishedApi\n@SinceKotlin("1.3")\ninternal expect fun checkIndexOverflow(index: Int):
Int\n\n@PublishedApi\n@SinceKotlin("1.3")\ninternal expect fun checkCountOverflow(count: Int):
Int\n\n@PublishedApi\n@SinceKotlin("1.3")\ninternal fun throwIndexOverflow() { throw
ArithmeticException("\nIndex overflow has happened.") }\n\n@PublishedApi\n@SinceKotlin("1.3")\ninternal fun
throwCountOverflow() { throw ArithmeticException("\nCount overflow has happened.") }\n\n\n"/*\n * Copyright
2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed
by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("\nMapsKt")\n@file:OptIn(kotlin.experiment
al.ExperimentalTypeInference::class)\n\npackage kotlin.collections\n\nimport kotlin.contracts.*\n\nprivate object
EmptyMap : Map<Any?, Nothing>, Serializable {\n    private const val serialVersionUID: Long =
8246714829545688274\n\n    override fun equals(other: Any?): Boolean = other is Map<*, *> &&
other.isEmpty()\n\n    override fun hashCode(): Int = 0\n\n    override fun toString(): String = "{}"\n\n    override val
size: Int get() = 0\n\n    override fun isEmpty(): Boolean = true\n\n    override fun containsKey(key: Any?): Boolean =
false\n\n    override fun containsValue(value: Nothing): Boolean = false\n\n    override fun get(key: Any?): Nothing? =
null\n\n    override val entries: Set<Map.Entry<Any?, Nothing>> get() = EmptySet\n\n    override val keys: Set<Any?>

```

```

get() = EmptySet\n  override val values: Collection<Nothing> get() = EmptyList\n\n  private fun readResolve():
Any = EmptyMap\n}\n\n/**\n * Returns an empty read-only map of specified type.\n *\n * The returned map is
serializable (JVM).\n * @sample samples.collections.Maps.Instantiation.emptyReadOnlyMap\n */\npublic fun <K,
V> emptyMap(): Map<K, V> = @Suppress("UNCHECKED_CAST") (EmptyMap as Map<K, V>)\n\n/**\n *
Returns a new read-only map with the specified contents, given as a list of pairs\n * where the first value is the key
and the second is the value.\n *\n * If multiple pairs have the same key, the resulting map will contain the value
from the last of those pairs.\n *\n * Entries of the map are iterated in the order they were specified.\n *\n * The
returned map is serializable (JVM).\n *\n * @sample samples.collections.Maps.Instantiation.mapFromPairs\n */\npublic fun <K, V> mapOf(vararg pairs: Pair<K, V>): Map<K, V> =\n  if (pairs.size > 0)
pairs.toMap(LinkedHashMap(mapCapacity(pairs.size))) else emptyMap()\n\n/**\n * Returns an empty read-only
map.\n *\n * The returned map is serializable (JVM).\n * @sample
samples.collections.Maps.Instantiation.emptyReadOnlyMap\n */\n@kotlin.internal.InlineOnly\npublic inline fun
<K, V> mapOf(): Map<K, V> = emptyMap()\n\n/**\n * Returns an empty new [MutableMap].\n *\n * The returned
map preserves the entry iteration order.\n * @sample samples.collections.Maps.Instantiation.emptyMutableMap\n */\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline fun <K, V> mutableMapOf():
MutableMap<K, V> = LinkedHashMap()\n\n/**\n * Returns a new [MutableMap] with the specified contents, given
as a list of pairs\n * where the first component is the key and the second is the value.\n *\n * If multiple pairs have
the same key, the resulting map will contain the value from the last of those pairs.\n *\n * Entries of the map are
iterated in the order they were specified.\n *\n * @sample
samples.collections.Maps.Instantiation.mutableMapFromPairs\n * @sample
samples.collections.Maps.Instantiation.emptyMutableMap\n */\npublic fun <K, V> mutableMapOf(vararg pairs:
Pair<K, V>): MutableMap<K, V> =\n  LinkedHashMap<K, V>(mapCapacity(pairs.size)).apply { putAll(pairs)
}\n\n/**\n * Returns an empty new [HashMap].\n *\n * @sample
samples.collections.Maps.Instantiation.emptyHashMap\n */\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline fun <K, V> hashMapOf(): HashMap<K, V>
= HashMap<K, V>()\n\n/**\n * Returns a new [HashMap] with the specified contents, given as a list of pairs\n *
where the first component is the key and the second is the value.\n *\n * @sample
samples.collections.Maps.Instantiation.hashMapFromPairs\n */\npublic fun <K, V> hashMapOf(vararg pairs:
Pair<K, V>): HashMap<K, V> = HashMap<K, V>(mapCapacity(pairs.size)).apply { putAll(pairs) }\n\n/**\n *
Returns an empty new [LinkedHashMap].\n *\n * @sample samples.collections.Maps.Instantiation.emptyHashMap\n */\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline
fun <K, V> linkedMapOf(): LinkedHashMap<K, V> = LinkedHashMap<K, V>()\n\n/**\n * Returns a new
[LinkedHashMap] with the specified contents, given as a list of pairs\n * where the first component is the key and
the second is the value.\n *\n * If multiple pairs have the same key, the resulting map will contain the value from the
last of those pairs.\n *\n * Entries of the map are iterated in the order they were specified.\n *\n * @sample
samples.collections.Maps.Instantiation.linkedMapFromPairs\n */\npublic fun <K, V> linkedMapOf(vararg pairs:
Pair<K, V>): LinkedHashMap<K, V> = pairs.toMap(LinkedHashMap(mapCapacity(pairs.size)))\n\n/**\n * Builds
a new read-only [Map] by populating a [MutableMap] using the given [builderAction]\n * and returning a read-only
map with the same key-value pairs.\n *\n * The map passed as a receiver to the [builderAction] is valid only inside
that function.\n * Using it outside of the function produces an unspecified behavior.\n *\n * Entries of the map are
iterated in the order they were added by the [builderAction].\n *\n * The returned map is serializable (JVM).\n *\n *
@sample samples.collections.Builders.Maps.buildMapSample\n */\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun <K, V> buildMap(@BuilderInference builderAction: MutableMap<K, V>().->Unit): Map<K, V> {\n  \n  contract { callsInPlace(builderAction, InvocationKind.EXACTLY_ONCE) }\n  \n  return
buildMapInternal(builderAction)\n}\n\n@PublishedApi\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\ninternal
expect inline fun <K, V> buildMapInternal(builderAction: MutableMap<K, V>().->Unit): Map<K,
V>\n\n/**\n * Builds a new read-only [Map] by populating a [MutableMap] using the given [builderAction]\n * and
returning a read-only map with the same key-value pairs.\n *\n * The map passed as a receiver to the

```

[builderAction] is valid only inside that function.  
 \* Using it outside of the function produces an unspecified behavior.  
 \* [capacity] is used to hint the expected number of pairs added in the [builderAction].  
 \* Entries of the map are iterated in the order they were added by the [builderAction].  
 \* The returned map is serializable (JVM).  
 \* @throws IllegalArgumentException if the given [capacity] is negative.  
 \* @sample samples.collections.Builders.Maps.buildMapSample

```

*^@SinceKotlin("1.6")^@WasExperimental(ExperimentalStdlibApi::class)^@kotlin.internal.InlineOnly^public
c inline fun <K, V> buildMap(capacity: Int, @BuilderInference builderAction: MutableMap<K, V>().->Unit):
Map<K, V> {
    contract { callsInPlace(builderAction, InvocationKind.EXACTLY_ONCE) }
    return
    buildMapInternal(capacity,
    builderAction)
}
^@PublishedApi^@SinceKotlin("1.3")^@kotlin.internal.InlineOnly^internal expect inline
fun <K, V> buildMapInternal(capacity: Int, builderAction: MutableMap<K, V>().->Unit): Map<K, V> {
    // Calculate the initial capacity of a map.
    // ^@PublishedApi^internal expect fun mapCapacity(expectedSize: Int):
    Int {
        // Returns `true` if this map is not empty.
        // @sample
        samples.collections.Maps.Usage.mapIsNotEmpty
    }
    // ^@kotlin.internal.InlineOnly^public inline fun <K, V>
    Map<out K, V>.isEmpty(): Boolean = !isNotEmpty()
    // Returns `true` if this nullable map is either null or
    empty.
    // @sample samples.collections.Maps.Usage.mapIsNullOrEmpty
}
*^@SinceKotlin("1.3")^@kotlin.internal.InlineOnly^public inline fun <K, V> Map<out K,
V>?.isNullOrEmpty(): Boolean {
    contract {
        returns(false) implies (this@isNullOrEmpty != null)
    }
    return this == null || isEmpty()
}
*^@sample samples.collections.Maps.Usage.mapOrEmpty
*^@kotlin.internal.InlineOnly^public inline fun
<K, V> Map<K, V>?.orEmpty(): Map<K, V> = this ?: emptyMap()
*^@sample
samples.collections.Maps.Usage.mapIfEmpty
*^@SinceKotlin("1.3")^@kotlin.internal.InlineOnly^public
inline fun <M, R> M.ifEmpty(defaultValue: () -> R): R where M : Map<*, *>, M : R =
    if (isEmpty())
    defaultValue() else this
*^@sample
samples.collections.Maps.Usage.containsKey
*^@kotlin.internal.InlineOnly^public inline operator fun
<@kotlin.internal.OnlyInputTypes K, V> Map<out K, V>.contains(key: K): Boolean =
    containsKey(key)
*^@sample
Returns the value corresponding to the given [key], or `null` if such a key is not present in the map.
*^@kotlin.internal.InlineOnly^public inline operator fun <@kotlin.internal.OnlyInputTypes K, V> Map<out K,
V>.get(key: K): V? =
    @Suppress("UNCHECKED_CAST") (this as Map<K, V>).get(key)
*^@sample
Allows to use the index operator for storing values in a mutable map.
*^@kotlin.internal.InlineOnly^public inline
operator fun <K, V> MutableMap<K, V>.set(key: K, value: V): Unit {
    put(key, value)
}
*^@sample
Returns `true` if the map contains the specified [key].
*^@sample
Allows to overcome type-safety restriction of `containsKey`
that requires to pass a key of type `K`.
*^@kotlin.internal.InlineOnly^public inline fun
<@kotlin.internal.OnlyInputTypes K> Map<out K, *>.containsKey(key: K): Boolean =
    @Suppress("UNCHECKED_CAST") (this as Map<K, *>).containsKey(key)
*^@sample
Returns `true` if the map
maps one or more keys to the specified [value].
*^@sample
Allows to overcome type-safety restriction of
`containsValue` that requires to pass a value of type `V`.
*^@sample
samples.collections.Maps.Usage.containsValue
*^@Suppress("EXTENSION_SHADOWED_BY_MEMBER")
// false warning, extension takes precedence in some cases
*^@kotlin.internal.InlineOnly^public inline fun <K,
@kotlin.internal.OnlyInputTypes V> Map<K, V>.containsValue(value: V): Boolean =
    this.containsValue(value)
*^@sample
Removes the specified key and its corresponding value from this map.
*^@sample
@return the previous value associated with the key, or `null` if the key was not present in the map.
*^@sample
Allows to
overcome type-safety restriction of `remove` that requires to pass a key of type `K`.
*^@kotlin.internal.InlineOnly^public inline fun <@kotlin.internal.OnlyInputTypes K, V> MutableMap<out K,
V>.remove(key: K): V? =
    @Suppress("UNCHECKED_CAST") (this as MutableMap<K,
V>).remove(key)
*^@sample
Returns the key component of the map entry.
*^@sample
This method allows to use

```

destructuring declarations when working with maps, for example:

```

for ((key, value) in map) {
    // do something with the key and the value
}

```

`@kotlin.internal.InlineOnly` public inline operator fun <K, V> Map.Entry<K, V>.component1(): K = key

Returns the value component of the map entry.

This method allows to use destructuring declarations when working with maps, for example:

```

for ((key, value) in map) {
    // do something with the key and the value
}

```

`@kotlin.internal.InlineOnly` public inline operator fun <K, V> Map.Entry<K, V>.component2(): V = value

Converts entry to [Pair] with key being first component and value being second.

`@kotlin.internal.InlineOnly` public inline fun <K, V> Map.Entry<K, V>.toPair(): Pair<K, V> = Pair(key, value)

Returns the value for the given key, or the result of the [defaultValue] function if there was no entry for the given key.

`@sample` samples.collections.Maps.Usage.getOrElse

`@kotlin.internal.InlineOnly` public inline fun <K, V> Map<K, V>.getOrElse(key: K, defaultValue: () -> V): V = get(key) ?: defaultValue

internal inline fun <K, V> Map<K, V>.getOrElseNullable(key: K, defaultValue: () -> V): V {
 val value = get(key)
 if (value == null && !containsKey(key)) {
 return defaultValue()
 } else {
 @Suppress("UNCHECKED\_CAST")
 return value as V
 }
}

Returns the value for the given [key] or throws an exception if there is no such key in the map.

If the map was created by [withDefault], resorts to its `defaultValue` provider function instead of throwing an exception.

`@throws` NoSuchElementException when the map doesn't contain a value for the specified key and no implicit default value was provided for that map.

`@SinceKotlin("1.1")` public fun <K, V> Map<K, V>.getValue(key: K): V = getOrElse(key, defaultValue)

Returns the value for the given key. If the key is not found in the map, calls the [defaultValue] function, puts its result into the map under the given key and returns it.

Note that the operation is not guaranteed to be atomic if the map is being modified concurrently.

`@sample` samples.collections.Maps.Usage.getOrPut

`@kotlin.internal.InlineOnly` public inline fun <K, V> MutableMap<K, V>.getOrPut(key: K, defaultValue: () -> V): V {
 val value = get(key)
 return if (value == null) {
 val answer = defaultValue()
 put(key, answer)
 answer
 } else {
 value
 }
}

Returns an [Iterator] over the entries in the [Map].

`@sample` samples.collections.Maps.Usage.forOverEntries

`@kotlin.internal.InlineOnly` public inline operator fun <K, V> Map<out K, V>.iterator(): Iterator<Map.Entry<K, V>> = entries.iterator()

Returns a [MutableIterator] over the mutable entries in the [MutableMap].

`@kotlin.jvm.JvmName("mutableIterator")` `@kotlin.internal.InlineOnly` public inline operator fun <K, V> MutableMap<K, V>.iterator(): MutableIterator<MutableMap.MutableEntry<K, V>> = entries.iterator()

Populates the given [destination] map with entries having the keys of this map and the values obtained by applying the [transform] function to each entry in this [Map].

`@kotlin.internal.InlineOnly` public inline fun <K, V, R, M : MutableMap<in K, in R>> Map<out K, V>.mapValuesTo(destination: M, transform: (Map.Entry<K, V>) -> R): M {
 return entries.associateByTo(destination, { it.key }, transform)
}

Populates the given [destination] map with entries having the keys obtained by applying the [transform] function to each entry in this [Map] and the values of this map.

In case if any two entries are mapped to the equal keys, the value of the latter one will overwrite the value associated with the former one.

`@kotlin.internal.InlineOnly` public inline fun <K, V, R, M : MutableMap<in R, in V>> Map<out K, V>.mapKeysTo(destination: M, transform: (Map.Entry<K, V>) -> R): M {
 return entries.associateByTo(destination, transform, { it.value })
}

Puts all the given [pairs] into this [MutableMap] with the first component in the pair being the key and the second the value.

`@kotlin.internal.InlineOnly` public fun <K, V> MutableMap<in K, in V>.putAll(pairs: Array<out Pair<K, V>>): Unit {
 for ((key, value) in pairs) {
 put(key, value)
 }
}

Puts all the elements of the given collection into this [MutableMap] with the first component in the pair being the key and the second the value.

`@kotlin.internal.InlineOnly` public fun <K, V> MutableMap<in K, in V>.putAll(pairs: Iterable<Pair<K, V>>): Unit {
 for ((key, value) in pairs) {
 put(key, value)
 }
}

Puts all the elements of the given sequence into this [MutableMap] with the first component in the pair being the key and the second the value.

`@kotlin.internal.InlineOnly` public fun <K, V> MutableMap<in K, in V>.putAll(pairs: Sequence<Pair<K, V>>): Unit {
 for ((key, value) in pairs) {
 put(key, value)
 }
}

Returns a new map with entries having the keys of this map and the values obtained by applying the [transform] function to each entry in this [Map].

The returned map preserves the entry iteration order of the original map.

```

@sample samples.collections.Maps.Transformations.mapValues\n *\npublic inline fun <K, V, R> Map<out K,
V>.mapValues(transform: (Map.Entry<K, V>) -> R): Map<K, R> {\n    return mapValuesTo(LinkedHashMap<K,
R>(mapCapacity(size)), transform) // .optimizeReadOnlyMap()\n}\n\n/**\n * Returns a new Map with entries
having the keys obtained by applying the [transform] function to each entry in this\n * [Map] and the values of this
map.\n * In case if any two entries are mapped to the equal keys, the value of the latter one will overwrite\n * the
value associated with the former one.\n * The returned map preserves the entry iteration order of the original
map.\n * @sample samples.collections.Maps.Transformations.mapKeys\n *\npublic inline fun <K, V, R>
Map<out K, V>.mapKeys(transform: (Map.Entry<K, V>) -> R): Map<R, V> {\n    return
mapKeysTo(LinkedHashMap<R, V>(mapCapacity(size)), transform) // .optimizeReadOnlyMap()\n}\n\n/**\n *
Returns a map containing all key-value pairs with keys matching the given [predicate].\n * The returned map
preserves the entry iteration order of the original map.\n * @sample samples.collections.Maps.Filtering.filterKeys\n
*\npublic inline fun <K, V> Map<out K, V>.filterKeys(predicate: (K) -> Boolean): Map<K, V> {\n    val result =
LinkedHashMap<K, V>()\n    for (entry in this) {\n        if (predicate(entry.key)) {\n            result.put(entry.key,
entry.value)\n        }\n    }\n    return result\n}\n\n/**\n * Returns a map containing all key-value pairs with values
matching the given [predicate].\n * The returned map preserves the entry iteration order of the original map.\n *
@sample samples.collections.Maps.Filtering.filterValues\n *\npublic inline fun <K, V> Map<out K,
V>.filterValues(predicate: (V) -> Boolean): Map<K, V> {\n    val result = LinkedHashMap<K, V>()\n    for (entry
in this) {\n        if (predicate(entry.value)) {\n            result.put(entry.key, entry.value)\n        }\n    }\n    return
result\n}\n\n/**\n * Appends all entries matching the given [predicate] into the mutable map given as [destination]
parameter.\n * @return the destination map.\n * @sample samples.collections.Maps.Filtering.filterTo\n
*\npublic inline fun <K, V, M : MutableMap<in K, in V>> Map<out K, V>.filterTo(destination: M, predicate:
(Map.Entry<K, V>) -> Boolean): M {\n    for (element in this) {\n        if (predicate(element)) {\n
destination.put(element.key, element.value)\n        }\n    }\n    return destination\n}\n\n/**\n * Returns a new map
containing all key-value pairs matching the given [predicate].\n * The returned map preserves the entry iteration
order of the original map.\n * @sample samples.collections.Maps.Filtering.filter\n *\npublic inline fun <K, V>
Map<out K, V>.filter(predicate: (Map.Entry<K, V>) -> Boolean): Map<K, V> {\n    return
filterTo(LinkedHashMap<K, V>(), predicate)\n}\n\n/**\n * Appends all entries not matching the given [predicate]
into the given [destination].\n * @return the destination map.\n * @sample
samples.collections.Maps.Filtering.filterNotTo\n *\npublic inline fun <K, V, M : MutableMap<in K, in V>>
Map<out K, V>.filterNotTo(destination: M, predicate: (Map.Entry<K, V>) -> Boolean): M {\n    for (element in
this) {\n        if (!predicate(element)) {\n            destination.put(element.key, element.value)\n        }\n    }\n    return
destination\n}\n\n/**\n * Returns a new map containing all key-value pairs not matching the given [predicate].\n *
The returned map preserves the entry iteration order of the original map.\n * @sample
samples.collections.Maps.Filtering.filterNot\n *\npublic inline fun <K, V> Map<out K, V>.filterNot(predicate:
(Map.Entry<K, V>) -> Boolean): Map<K, V> {\n    return filterNotTo(LinkedHashMap<K, V>(),
predicate)\n}\n\n/**\n * Returns a new map containing all key-value pairs from the given collection of pairs.\n *
The returned map preserves the entry iteration order of the original collection.\n * If any of two pairs would have the
same key the last one gets added to the map.\n *\npublic fun <K, V> Iterable<Pair<K, V>>.toMap(): Map<K, V>
{\n    if (this is Collection) {\n        return when (size) {\n            0 -> emptyMap()\n            1 -> mapOf(if (this is
List) this[0] else iterator().next())\n            else -> toMap(LinkedHashMap<K, V>(mapCapacity(size)))\n        }\n    }\n    return toMap(LinkedHashMap<K, V>()).optimizeReadOnlyMap()\n}\n\n/**\n * Populates and returns the
[destination] mutable map with key-value pairs from the given collection of pairs.\n *\npublic fun <K, V, M :
MutableMap<in K, in V>> Iterable<Pair<K, V>>.toMap(destination: M): M =\n    destination.apply {
putAll(this@toMap) }\n\n/**\n * Returns a new map containing all key-value pairs from the given array of pairs.\n *
The returned map preserves the entry iteration order of the original array.\n * If any of two pairs would have
the same key the last one gets added to the map.\n *\npublic fun <K, V> Array<out Pair<K, V>>.toMap(): Map<K,
V> = when (size) {\n    0 -> emptyMap()\n    1 -> mapOf(this[0])\n    else -> toMap(LinkedHashMap<K,
V>(mapCapacity(size)))\n}\n\n/**\n * Populates and returns the [destination] mutable map with key-value pairs

```

from the given array of pairs.

```

public fun <K, V, M : MutableMap<in K, in V>> Array<out Pair<K, V>>.toMap(destination: M): M =\n destination.apply { putAll(this@toMap) }\n\n/**\n * Returns a new map containing all key-value pairs from the given sequence of pairs.\n * The returned map preserves the entry iteration order of the original sequence.\n * If any of two pairs would have the same key the last one gets added to the map.\n */\npublic fun <K, V> Sequence<Pair<K, V>>.toMap(): Map<K, V> = toMap(LinkedHashMap<K, V>()).optimizeReadOnlyMap()\n\n/**\n * Populates and returns the [destination] mutable map with key-value pairs from the given sequence of pairs.\n */\npublic fun <K, V, M : MutableMap<in K, in V>> Sequence<Pair<K, V>>.toMap(destination: M): M =\n destination.apply { putAll(this@toMap) }\n\n/**\n * Returns a new read-only map containing all key-value pairs from the original map.\n * The returned map preserves the entry iteration order of the original map.\n */\n@SinceKotlin("1.1")\npublic fun <K, V> Map<out K, V>.toMap(): Map<K, V> = when (size) {\n 0 -> emptyMap()\n 1 -> toSingletonMap()\n else -> toMutableMap()\n}\n\n/**\n * Returns a new mutable map containing all key-value pairs from the original map.\n * The returned map preserves the entry iteration order of the original map.\n */\n@SinceKotlin("1.1")\npublic fun <K, V> Map<out K, V>.toMutableMap(): MutableMap<K, V> = LinkedHashMap(this)\n\n/**\n * Populates and returns the [destination] mutable map with key-value pairs from the given map.\n */\n@SinceKotlin("1.1")\npublic fun <K, V, M : MutableMap<in K, in V>> Map<out K, V>.toMap(destination: M): M =\n destination.apply { putAll(this@toMap) }\n\n/**\n * Creates a new read-only map by replacing or adding an entry to this map from a given key-value [pair].\n * The returned map preserves the entry iteration order of the original map.\n * The [pair] is iterated in the end if it has a unique key.\n */\npublic operator fun <K, V> Map<out K, V>.plus(pair: Pair<K, V>): Map<K, V> =\n if (this.isEmpty()) mapOf(pair) else LinkedHashMap(this).apply { put(pair.first, pair.second) }\n\n/**\n * Creates a new read-only map by replacing or adding entries to this map from a given collection of key-value [pairs].\n * The returned map preserves the entry iteration order of the original map.\n * Those [pairs] with unique keys are iterated in the end in the order of [pairs] collection.\n */\npublic operator fun <K, V> Map<out K, V>.plus(pairs: Iterable<Pair<K, V>>): Map<K, V> =\n if (this.isEmpty()) pairs.toMap() else LinkedHashMap(this).apply { putAll(pairs) }\n\n/**\n * Creates a new read-only map by replacing or adding entries to this map from a given array of key-value [pairs].\n * The returned map preserves the entry iteration order of the original map.\n * Those [pairs] with unique keys are iterated in the end in the order of [pairs] array.\n */\npublic operator fun <K, V> Map<out K, V>.plus(pairs: Array<out Pair<K, V>>): Map<K, V> =\n if (this.isEmpty()) pairs.toMap() else LinkedHashMap(this).apply { putAll(pairs) }\n\n/**\n * Creates a new read-only map by replacing or adding entries to this map from a given sequence of key-value [pairs].\n * The returned map preserves the entry iteration order of the original map.\n * Those [pairs] with unique keys are iterated in the end in the order of [pairs] sequence.\n */\npublic operator fun <K, V> Map<out K, V>.plus(pairs: Sequence<Pair<K, V>>): Map<K, V> =\n LinkedHashMap(this).apply { putAll(pairs) }.optimizeReadOnlyMap()\n\n/**\n * Creates a new read-only map by replacing or adding entries to this map from another [map].\n * The returned map preserves the entry iteration order of the original map.\n * Those entries of another [map] that are missing in this map are iterated in the end in the order of that [map].\n */\npublic operator fun <K, V> Map<out K, V>.plus(map: Map<out K, V>): Map<K, V> =\n LinkedHashMap(this).apply { putAll(map) }\n\n/**\n * Appends or replaces the given [pair] in this mutable map.\n */\n@kotlin.internal.InlineOnly\npublic inline operator fun <K, V> MutableMap<in K, in V>.plusAssign(pair: Pair<K, V>) {\n put(pair.first, pair.second)\n}\n\n/**\n * Appends or replaces all pairs from the given collection of [pairs] in this mutable map.\n */\n@kotlin.internal.InlineOnly\npublic inline operator fun <K, V> MutableMap<in K, in V>.plusAssign(pairs: Iterable<Pair<K, V>>) {\n putAll(pairs)\n}\n\n/**\n * Appends or replaces all pairs from the given array of [pairs] in this mutable map.\n */\n@kotlin.internal.InlineOnly\npublic inline operator fun <K, V> MutableMap<in K, in V>.plusAssign(pairs: Array<out Pair<K, V>>) {\n putAll(pairs)\n}\n\n/**\n * Appends or replaces all pairs from the given sequence of [pairs] in this mutable map.\n */\n@kotlin.internal.InlineOnly\npublic inline operator fun <K, V> MutableMap<in K, in V>.plusAssign(pairs: Sequence<Pair<K, V>>) {\n putAll(pairs)\n}\n\n/**\n * Appends or replaces all entries from the given [map] in this mutable map.\n */\n@kotlin.internal.InlineOnly\npublic inline operator fun <K, V> MutableMap<in K, in V>.plusAssign(map: Map<K, V>) {\n putAll(map)\n}\n\n/**\n * Returns a map

```

containing all entries of the original map except the entry with the given [key].

`fun <K, V> Map<out K, V>.minus(key: K): Map<K, V> =` this.toMutableMap().apply { minusAssign(key) }.optimizeReadOnlyMap()

`fun <K, V> Map<out K, V>.minus(keys: Iterable<K>): Map<K, V> =` this.toMutableMap().apply { minusAssign(keys) }.optimizeReadOnlyMap()

`fun <K, V> Map<out K, V>.minus(keys: Array<out K>): Map<K, V> =` this.toMutableMap().apply { minusAssign(keys) }.optimizeReadOnlyMap()

`fun <K, V> Map<out K, V>.minus(keys: Sequence<K>): Map<K, V> =` this.toMutableMap().apply { minusAssign(keys) }.optimizeReadOnlyMap()

`fun <K, V> MutableMap<K, V>.minusAssign(key: K) {` remove(key)

`fun <K, V> MutableMap<K, V>.minusAssign(keys: Iterable<K>) {` this.keys.removeAll(keys)

`fun <K, V> MutableMap<K, V>.minusAssign(keys: Array<out K>) {` this.keys.removeAll(keys)

`fun <K, V> MutableMap<K, V>.minusAssign(keys: Sequence<K>) {` this.keys.removeAll(keys)

`@PublishedApi internal fun <K, V> Map<K, V>.optimizeReadOnlyMap() =` when (size) { 0 -> emptyMap() 1 -> toSingletonMapOrSelf() else -> this }

`/* Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors. Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.`

`@file:kotlin.jvm.JvmMultifileClass @file:kotlin.jvm.JvmName("SetsKt") @file:OptIn(kotlin.experimental.ExperimentalTypeInference::class)`

`package kotlin.collections`

`import kotlin.contracts.*`

`internal object EmptySet : Set<Nothing>, Serializable {`

`private const val serialVersionUID: Long = 3406603774387020532`

`override fun equals(other: Any?): Boolean = other is Set<*> && other.isEmpty()`

`override fun hashCode(): Int = 0`

`override fun toString(): String = "[]"`

`override val size: Int get() = 0`

`override fun isEmpty(): Boolean = true`

`override fun contains(element: Nothing): Boolean = false`

`override fun containsAll(elements: Collection<Nothing>): Boolean = elements.isEmpty()`

`override fun iterator(): Iterator<Nothing> = EmptyIterator`

`private fun readResolve(): Any = EmptySet`

`/* Returns an empty read-only set. The returned set is serializable (JVM). */`

`@sample`

`samples.collections.Collections.Sets.emptyReadOnlySet`

`fun <T> emptySet(): Set<T> = EmptySet`

`/* Returns a new read-only set with the given elements. Elements of the set are iterated in the order they were specified. The returned set is serializable (JVM). */`

`@sample`

`samples.collections.Collections.Sets.readOnlySet`

`fun <T> setOf(vararg elements: T): Set<T> = if (elements.size > 0) elements.toSet() else emptySet()`

`/* Returns an empty read-only set. The returned set is serializable (JVM). */`

`@sample`

`samples.collections.Collections.Sets.emptyReadOnlySet`

`fun <T> setOf(): Set<T> = emptySet()`

`/* Returns an empty new [MutableSet]. The returned set preserves the element iteration order. */`

`@sample`

`samples.collections.Collections.Sets.emptyMutableSet`  
`*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline fun <T> mutableSetOf(): MutableSet<T> =`  
`LinkedHashSet()\n\n/**\n * Returns a new [MutableSet] with the given elements.\n * Elements of the set are`  
`iterated in the order they were specified.\n * @sample samples.collections.Collections.Sets.mutableSet\n */\npublic`  
`fun <T> mutableSetOf(vararg elements: T): MutableSet<T> =`  
`elements.toCollection(LinkedHashSet(mapCapacity(elements.size)))\n\n/**\n * Returns an empty new [HashSet].`  
`*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline fun <T> hashSetOf(): HashSet<T> =`  
`HashSet()\n\n/**\n * Returns a new [HashSet] with the given elements. *\npublic fun <T> hashSetOf(vararg elements:`  
`T): HashSet<T> = elements.toCollection(HashSet(mapCapacity(elements.size)))\n\n/**\n * Returns an empty new`  
`[LinkedHashSet].\n * @sample samples.collections.Collections.Sets.emptyLinkedHashSet\n`  
`*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline fun <T> linkedSetOf(): LinkedHashSet<T>`  
`= LinkedHashSet()\n\n/**\n * Returns a new [LinkedHashSet] with the given elements.\n * Elements of the set are`  
`iterated in the order they were specified.\n * @sample samples.collections.Collections.Sets.linkedHashSet\n`  
`*\npublic fun <T> linkedSetOf(vararg elements: T): LinkedHashSet<T> =`  
`elements.toCollection(LinkedHashSet(mapCapacity(elements.size)))\n\n/**\n * Returns a new read-only set either`  
`with single given element, if it is not null, or empty set if the element is null.\n * The returned set is serializable`  
`(JVM).\n * @sample samples.collections.Collections.Sets.setOfNotNull\n */\n@SinceKotlin("1.4")\npublic fun <T`  
`: Any> setOfNotNull(element: T?): Set<T> = if (element != null) setOf(element) else emptySet()\n\n/**\n * Returns`  
`a new read-only set only with those given elements, that are not null.\n * Elements of the set are iterated in the order`  
`they were specified.\n * The returned set is serializable (JVM).\n * @sample`  
`samples.collections.Collections.Sets.setOfNotNull\n */\n@SinceKotlin("1.4")\npublic fun <T : Any>`  
`setOfNotNull(vararg elements: T?): Set<T> {\n return elements.filterNotNullTo(LinkedHashSet())\n}\n\n/**\n * Builds a new read-only [Set] by populating a [MutableSet] using the given [builderAction]\n * and returning a read-`  
`only set with the same elements.\n * \n * The set passed as a receiver to the [builderAction] is valid only inside that`  
`function.\n * Using it outside of the function produces an unspecified behavior.\n * \n * Elements of the set are`  
`iterated in the order they were added by the [builderAction].\n * \n * The returned set is serializable (JVM).\n * \n *`  
`@sample samples.Collections.Builders.Sets.buildSetSample\n`  
`*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic`  
`inline fun <E> buildSet(@BuilderInference builderAction: MutableSet<E>().->Unit): Set<E> {\n contract {\n`  
`callsInPlace(builderAction, InvocationKind.EXACTLY_ONCE) }\n return`  
`buildSetInternal(builderAction)\n}\n\n@PublishedApi\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\ninternal`  
`expect inline fun <E> buildSetInternal(builderAction: MutableSet<E>().->Unit): Set<E>\n\n/**\n * Builds a`  
`new read-only [Set] by populating a [MutableSet] using the given [builderAction]\n * and returning a read-only set`  
`with the same elements.\n * \n * The set passed as a receiver to the [builderAction] is valid only inside that`  
`function.\n * Using it outside of the function produces an unspecified behavior.\n * \n * [capacity] is used to hint the`  
`expected number of elements added in the [builderAction].\n * \n * Elements of the set are iterated in the order they`  
`were added by the [builderAction].\n * \n * The returned set is serializable (JVM).\n * \n * @throws`  
`IllegalArgumentException if the given [capacity] is negative.\n * \n * @sample`  
`samples.Collections.Builders.Sets.buildSetSample\n`  
`*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic`  
`inline fun <E> buildSet(capacity: Int, @BuilderInference builderAction: MutableSet<E>().->Unit): Set<E> {\n`  
`contract {\n callsInPlace(builderAction, InvocationKind.EXACTLY_ONCE) }\n return buildSetInternal(capacity,`  
`builderAction)\n}\n\n@PublishedApi\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\ninternal expect inline`  
`fun <E> buildSetInternal(capacity: Int, builderAction: MutableSet<E>().->Unit): Set<E>\n\n/**\n * Returns this Set`  
`if it's not `null` and the empty set otherwise. *\n@kotlin.internal.InlineOnly\npublic inline fun <T>`  
`Set<T>?.orEmpty(): Set<T> = this ?: emptySet()\n\ninternal fun <T> Set<T>.optimizeReadOnlySet() = when (size)`  
`{\n 0 -> emptySet()\n 1 -> setOf(iterator().next())\n else -> this\n}\n\n"/*\n * Copyright 2010-2018 JetBrains`  
`s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0`



```

return if (isNegative) result else -result\n}\n\ninternal fun numberFormatError(input: String): Nothing = throw
NumberFormatException("Invalid number format: '$input')\n", "/*\n * Copyright 2010-2021 JetBrains s.r.o. and
Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.time\n\nimport kotlin.contracts.*\nimport
kotlin.jvm.JvmInline\nimport kotlin.math.*\n\n/**\n * Represents the amount of time one instant of time is away
from another instant.\n * A negative duration is possible in a situation when the second instant is earlier than the
first one.\n * The type can store duration values up to \u00b1146 years with nanosecond precision,\n * and up to
\u00b1146 million years with millisecond precision.\n * If a duration-returning operation provided in `kotlin.time`
produces a duration value that doesn't fit into the above range,\n * the returned `Duration` is infinite.\n * An
infinite duration value [Duration.INFINITE] can be used to represent infinite timeouts.\n * To construct a
duration use either the extension function [toDuration],\n * or the extension properties [hours], [minutes], [seconds],
and so on,\n * available on [Int], [Long], and [Double] numeric types.\n * To get the value of this duration
expressed in a particular [duration units][DurationUnit]\n * use the functions [toInt], [toLong], and [toDouble]\n * or
the properties [inWholeHours], [inWholeMinutes], [inWholeSeconds], [inWholeNanoseconds], and so on.\n
*/\n\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalTime::class)\n@JvmInline\npublic value class
Duration internal constructor(private val rawValue: Long) : Comparable<Duration> {\n\n    private val value: Long
get() = rawValue shr 1\n    private inline val unitDiscriminator: Int get() = rawValue.toInt() and 1\n    private fun
isInNanos() = unitDiscriminator == 0\n    private fun isInMillis() = unitDiscriminator == 1\n    private val
storageUnit get() = if (isInNanos()) DurationUnit.NANOSECONDS else DurationUnit.MILLISECONDS\n\n    init
{\n        if (durationAssertionsEnabled) {\n            if (isInNanos()) {\n                if (value !in -
MAX_NANOS..MAX_NANOS) throw AssertionError("\$value ns is out of nanoseconds range")\n            } else
{\n                if (value !in -MAX_MILLIS..MAX_MILLIS) throw AssertionError("\$value ms is out of milliseconds
range")\n                if (value in -MAX_NANOS_IN_MILLIS..MAX_NANOS_IN_MILLIS) throw
AssertionError("\$value ms is denormalized")\n            }\n        }\n    }\n\n    companion object {\n        /** The
duration equal to exactly 0 seconds. */\n        public val ZERO: Duration = Duration(0L)\n\n        /** The duration
whose value is positive infinity. It is useful for representing timeouts that should never expire. */\n        public val
INFINITE: Duration = durationOfMillis(MAX_MILLIS)\n\n        internal val NEG_INFINITE: Duration =
durationOfMillis(-MAX_MILLIS)\n\n        /** Converts the given time duration [value] expressed in the specified
[sourceUnit] into the specified [targetUnit]. */\n        @ExperimentalTime\n        public fun convert(value: Double,
sourceUnit: DurationUnit, targetUnit: DurationUnit): Double =\n            convertDurationUnit(value, sourceUnit,
targetUnit)\n\n        // Duration construction extension properties in Duration companion scope\n\n        /** Returns a
[Duration] equal to this [Int] number of nanoseconds. */\n        @kotlin.internal.InlineOnly\n        public inline val
Int.nanoseconds get() = toDuration(DurationUnit.NANOSECONDS)\n\n        /** Returns a [Duration] equal to this
[Long] number of nanoseconds. */\n        @kotlin.internal.InlineOnly\n        public inline val Long.nanoseconds
get() = toDuration(DurationUnit.NANOSECONDS)\n\n        /**\n         * Returns a [Duration] equal to this
[Double] number of nanoseconds.\n         *\n         * Depending on its magnitude, the value is rounded to an integer
number of nanoseconds or milliseconds.\n         *\n         * @throws IllegalArgumentException if this [Double]
value is `NaN`.\n         */\n        @kotlin.internal.InlineOnly\n        public inline val Double.nanoseconds get() =
toDuration(DurationUnit.NANOSECONDS)\n\n        /** Returns a [Duration] equal to this [Int] number of
microseconds. */\n        @kotlin.internal.InlineOnly\n        public inline val Int.microseconds get() =
toDuration(DurationUnit.MICROSECONDS)\n\n        /** Returns a [Duration] equal to this [Long] number of
microseconds. */\n        @kotlin.internal.InlineOnly\n        public inline val Long.microseconds get() =
toDuration(DurationUnit.MICROSECONDS)\n\n        /**\n         * Returns a [Duration] equal to this [Double]
number of microseconds.\n         *\n         * Depending on its magnitude, the value is rounded to an integer number
of nanoseconds or milliseconds.\n         *\n         * @throws IllegalArgumentException if this [Double] value is
`NaN`.\n         */\n        @kotlin.internal.InlineOnly\n        public inline val Double.microseconds get() =
toDuration(DurationUnit.MICROSECONDS)\n\n        /** Returns a [Duration] equal to this [Int] number of
milliseconds. */\n        @kotlin.internal.InlineOnly\n        public inline val Int.milliseconds get() =

```

```

toDuration(DurationUnit.MILLISECONDS)\n\n    /** Returns a [Duration] equal to this [Long] number of
milliseconds. */\n    @kotlin.internal.InlineOnly\n    public inline val Long.milliseconds get() =
toDuration(DurationUnit.MILLISECONDS)\n\n    /**\n    * Returns a [Duration] equal to this [Double]
number of milliseconds.\n    *\n    * Depending on its magnitude, the value is rounded to an integer number of
nanoseconds or milliseconds.\n    *\n    * @throws IllegalArgumentException if this [Double] value is
`NaN`.\n    */\n    @kotlin.internal.InlineOnly\n    public inline val Double.milliseconds get() =
toDuration(DurationUnit.MILLISECONDS)\n\n    /** Returns a [Duration] equal to this [Int] number of
seconds. */\n    @kotlin.internal.InlineOnly\n    public inline val Int.seconds get() =
toDuration(DurationUnit.SECONDS)\n\n    /** Returns a [Duration] equal to this [Long] number of seconds. */\n
    @kotlin.internal.InlineOnly\n    public inline val Long.seconds get() =
toDuration(DurationUnit.SECONDS)\n\n    /**\n    * Returns a [Duration] equal to this [Double] number of
seconds.\n    *\n    * Depending on its magnitude, the value is rounded to an integer number of nanoseconds or
milliseconds.\n    *\n    * @throws IllegalArgumentException if this [Double] value is `NaN`.\n    */\n
    @kotlin.internal.InlineOnly\n    public inline val Double.seconds get() =
toDuration(DurationUnit.SECONDS)\n\n    /** Returns a [Duration] equal to this [Int] number of minutes. */\n
    @kotlin.internal.InlineOnly\n    public inline val Int.minutes get() = toDuration(DurationUnit.MINUTES)\n\n
    /** Returns a [Duration] equal to this [Long] number of minutes. */\n    @kotlin.internal.InlineOnly\n
    public inline val Long.minutes get() = toDuration(DurationUnit.MINUTES)\n\n    /**\n    * Returns a
[Duration] equal to this [Double] number of minutes.\n    *\n    * Depending on its magnitude, the value is
rounded to an integer number of nanoseconds or milliseconds.\n    *\n    * @throws IllegalArgumentException
if this [Double] value is `NaN`.\n    */\n    @kotlin.internal.InlineOnly\n    public inline val Double.minutes
get() = toDuration(DurationUnit.MINUTES)\n\n    /** Returns a [Duration] equal to this [Int] number of hours.
*/\n    @kotlin.internal.InlineOnly\n    public inline val Int.hours get() = toDuration(DurationUnit.HOURS)\n\n
    /** Returns a [Duration] equal to this [Long] number of hours. */\n    @kotlin.internal.InlineOnly\n    public
inline val Long.hours get() = toDuration(DurationUnit.HOURS)\n\n    /**\n    * Returns a [Duration] equal to
this [Double] number of hours.\n    *\n    * Depending on its magnitude, the value is rounded to an integer
number of nanoseconds or milliseconds.\n    *\n    * @throws IllegalArgumentException if this [Double]
value is `NaN`.\n    */\n    @kotlin.internal.InlineOnly\n    public inline val Double.hours get() =
toDuration(DurationUnit.HOURS)\n\n    /** Returns a [Duration] equal to this [Int] number of days. */\n
    @kotlin.internal.InlineOnly\n    public inline val Int.days get() = toDuration(DurationUnit.DAYS)\n\n    /**
Returns a [Duration] equal to this [Long] number of days. */\n    @kotlin.internal.InlineOnly\n    public inline
val Long.days get() = toDuration(DurationUnit.DAYS)\n\n    /**\n    * Returns a [Duration] equal to this
[Double] number of days.\n    *\n    * Depending on its magnitude, the value is rounded to an integer number
of nanoseconds or milliseconds.\n    *\n    * @throws IllegalArgumentException if this [Double] value is
`NaN`.\n    */\n    @kotlin.internal.InlineOnly\n    public inline val Double.days get() =
toDuration(DurationUnit.DAYS)\n\n    // deprecated static factory functions\n    /** Returns a [Duration]
representing the specified [value] number of nanoseconds. */\n    @SinceKotlin("1.5")\n
    @ExperimentalTime\n    @Deprecated("Use 'Int.nanoseconds' extension property from Duration.Companion
instead.", ReplaceWith("value.nanoseconds", "kotlin.time.Duration.Companion.nanoseconds"))\n
    @DeprecatedSinceKotlin(warningSince = "1.6")\n    public fun nanoseconds(value: Int): Duration =
value.toDuration(DurationUnit.NANOSECONDS)\n\n    /** Returns a [Duration] representing the specified
[value] number of nanoseconds. */\n    @SinceKotlin("1.5")\n    @ExperimentalTime\n
    @Deprecated("Use 'Long.nanoseconds' extension property from Duration.Companion instead.",
ReplaceWith("value.nanoseconds", "kotlin.time.Duration.Companion.nanoseconds"))\n
    @DeprecatedSinceKotlin(warningSince = "1.6")\n    public fun nanoseconds(value: Long): Duration =
value.toDuration(DurationUnit.NANOSECONDS)\n\n    /**\n    * Returns a [Duration] representing the
specified [value] number of nanoseconds.\n    *\n    * @throws IllegalArgumentException if the provided
`Double` [value] is `NaN`.\n    */\n    @SinceKotlin("1.5")\n    @ExperimentalTime\n

```

```

@Deprecated("Use 'Double.nanoseconds' extension property from Duration.Companion instead."),
ReplaceWith("value.nanoseconds", "kotlin.time.Duration.Companion.nanoseconds"))\n
@DeprecatedSinceKotlin(warningSince = "1.6")\n    public fun nanoseconds(value: Double): Duration =
value.toDuration(DurationUnit.NANOSECONDS)\n\n    /** Returns a [Duration] representing the specified
[value] number of microseconds. */\n    @SinceKotlin("1.5")\n    @ExperimentalTime\n
@Deprecated("Use 'Int.microseconds' extension property from Duration.Companion instead."),
ReplaceWith("value.microseconds", "kotlin.time.Duration.Companion.microseconds"))\n
@DeprecatedSinceKotlin(warningSince = "1.6")\n    public fun microseconds(value: Int): Duration =
value.toDuration(DurationUnit.MICROSECONDS)\n\n    /** Returns a [Duration] representing the specified
[value] number of microseconds. */\n    @SinceKotlin("1.5")\n    @ExperimentalTime\n
@Deprecated("Use 'Long.microseconds' extension property from Duration.Companion instead."),
ReplaceWith("value.microseconds", "kotlin.time.Duration.Companion.microseconds"))\n
@DeprecatedSinceKotlin(warningSince = "1.6")\n    public fun microseconds(value: Long): Duration =
value.toDuration(DurationUnit.MICROSECONDS)\n\n    /**\n    * Returns a [Duration] representing the
specified [value] number of microseconds.\n    * @throws IllegalArgumentException if the provided
`Double` [value] is `NaN`.\n    */\n    @SinceKotlin("1.5")\n    @ExperimentalTime\n
@Deprecated("Use 'Double.microseconds' extension property from Duration.Companion instead."),
ReplaceWith("value.microseconds", "kotlin.time.Duration.Companion.microseconds"))\n
@DeprecatedSinceKotlin(warningSince = "1.6")\n    public fun microseconds(value: Double): Duration =
value.toDuration(DurationUnit.MICROSECONDS)\n\n    /** Returns a [Duration] representing the specified
[value] number of milliseconds. */\n    @SinceKotlin("1.5")\n    @ExperimentalTime\n
@Deprecated("Use 'Int.milliseconds' extension property from Duration.Companion instead."),
ReplaceWith("value.milliseconds", "kotlin.time.Duration.Companion.milliseconds"))\n
@DeprecatedSinceKotlin(warningSince = "1.6")\n    public fun milliseconds(value: Int): Duration =
value.toDuration(DurationUnit.MILLISECONDS)\n\n    /** Returns a [Duration] representing the specified
[value] number of milliseconds. */\n    @SinceKotlin("1.5")\n    @ExperimentalTime\n
@Deprecated("Use 'Long.milliseconds' extension property from Duration.Companion instead."),
ReplaceWith("value.milliseconds", "kotlin.time.Duration.Companion.milliseconds"))\n
@DeprecatedSinceKotlin(warningSince = "1.6")\n    public fun milliseconds(value: Long): Duration =
value.toDuration(DurationUnit.MILLISECONDS)\n\n    /**\n    * Returns a [Duration] representing the
specified [value] number of milliseconds.\n    * @throws IllegalArgumentException if the provided
`Double` [value] is `NaN`.\n    */\n    @SinceKotlin("1.5")\n    @ExperimentalTime\n
@Deprecated("Use 'Double.milliseconds' extension property from Duration.Companion instead."),
ReplaceWith("value.milliseconds", "kotlin.time.Duration.Companion.milliseconds"))\n
@DeprecatedSinceKotlin(warningSince = "1.6")\n    public fun milliseconds(value: Double): Duration =
value.toDuration(DurationUnit.MILLISECONDS)\n\n    /** Returns a [Duration] representing the specified
[value] number of seconds. */\n    @SinceKotlin("1.5")\n    @ExperimentalTime\n    @Deprecated("Use
'Int.seconds' extension property from Duration.Companion instead.", ReplaceWith("value.seconds",
"kotlin.time.Duration.Companion.seconds"))\n    @DeprecatedSinceKotlin(warningSince = "1.6")\n    public
fun seconds(value: Int): Duration = value.toDuration(DurationUnit.SECONDS)\n\n    /** Returns a [Duration]
representing the specified [value] number of seconds. */\n    @SinceKotlin("1.5")\n    @ExperimentalTime\n
    @Deprecated("Use 'Long.seconds' extension property from Duration.Companion instead.",
ReplaceWith("value.seconds", "kotlin.time.Duration.Companion.seconds"))\n
@DeprecatedSinceKotlin(warningSince = "1.6")\n    public fun seconds(value: Long): Duration =
value.toDuration(DurationUnit.SECONDS)\n\n    /**\n    * Returns a [Duration] representing the specified
[value] number of seconds.\n    * @throws IllegalArgumentException if the provided `Double` [value] is
`NaN`.\n    */\n    @SinceKotlin("1.5")\n    @ExperimentalTime\n    @Deprecated("Use
'Double.seconds' extension property from Duration.Companion instead.", ReplaceWith("value.seconds",

```

```

\kotlin.time.Duration.Companion.seconds\))\n    @DeprecatedSinceKotlin(warningSince = \"1.6\")\n    public
fun seconds(value: Double): Duration = value.toDuration(DurationUnit.SECONDS)\n\n    /** Returns a
[Duration] representing the specified [value] number of minutes. */\n    @SinceKotlin(\"1.5\")\n
@ExperimentalTime\n    @Deprecated(\"Use 'Int.minutes' extension property from Duration.Companion
instead.\", ReplaceWith(\"value.minutes\", \"kotlin.time.Duration.Companion.minutes\"))\n
@DeprecatedSinceKotlin(warningSince = \"1.6\")\n    public fun minutes(value: Int): Duration =
value.toDuration(DurationUnit.MINUTES)\n\n    /** Returns a [Duration] representing the specified [value]
number of minutes. */\n    @SinceKotlin(\"1.5\")\n    @ExperimentalTime\n    @Deprecated(\"Use
'Long.minutes' extension property from Duration.Companion instead.\", ReplaceWith(\"value.minutes\",
\"kotlin.time.Duration.Companion.minutes\"))\n    @DeprecatedSinceKotlin(warningSince = \"1.6\")\n    public
fun minutes(value: Long): Duration = value.toDuration(DurationUnit.MINUTES)\n\n    /**\n    * Returns a
[Duration] representing the specified [value] number of minutes.\n    *\n    * @throws
IllegalArgumentException if the provided `Double` [value] is `NaN`.\n    */\n    @SinceKotlin(\"1.5\")\n
@ExperimentalTime\n    @Deprecated(\"Use 'Double.minutes' extension property from Duration.Companion
instead.\", ReplaceWith(\"value.minutes\", \"kotlin.time.Duration.Companion.minutes\"))\n
@DeprecatedSinceKotlin(warningSince = \"1.6\")\n    public fun minutes(value: Double): Duration =
value.toDuration(DurationUnit.MINUTES)\n\n    /** Returns a [Duration] representing the specified [value]
number of hours. */\n    @SinceKotlin(\"1.5\")\n    @ExperimentalTime\n    @Deprecated(\"Use 'Int.hours'
extension property from Duration.Companion instead.\", ReplaceWith(\"value.hours\",
\"kotlin.time.Duration.Companion.hours\"))\n    @DeprecatedSinceKotlin(warningSince = \"1.6\")\n    public
fun hours(value: Int): Duration = value.toDuration(DurationUnit.HOURS)\n\n    /** Returns a [Duration]
representing the specified [value] number of hours. */\n    @SinceKotlin(\"1.5\")\n    @ExperimentalTime\n
@Deprecated(\"Use 'Long.hours' extension property from Duration.Companion instead.\",
ReplaceWith(\"value.hours\", \"kotlin.time.Duration.Companion.hours\"))\n
@DeprecatedSinceKotlin(warningSince = \"1.6\")\n    public fun hours(value: Long): Duration =
value.toDuration(DurationUnit.HOURS)\n\n    /**\n    * Returns a [Duration] representing the specified
[value] number of hours.\n    *\n    * @throws IllegalArgumentException if the provided `Double` [value] is
`NaN`.\n    */\n    @SinceKotlin(\"1.5\")\n    @ExperimentalTime\n    @Deprecated(\"Use 'Double.hours'
extension property from Duration.Companion instead.\", ReplaceWith(\"value.hours\",
\"kotlin.time.Duration.Companion.hours\"))\n    @DeprecatedSinceKotlin(warningSince = \"1.6\")\n    public
fun hours(value: Double): Duration = value.toDuration(DurationUnit.HOURS)\n\n    /** Returns a [Duration]
representing the specified [value] number of days. */\n    @SinceKotlin(\"1.5\")\n    @ExperimentalTime\n
@Deprecated(\"Use 'Int.days' extension property from Duration.Companion instead.\", ReplaceWith(\"value.days\",
\"kotlin.time.Duration.Companion.days\"))\n    @DeprecatedSinceKotlin(warningSince = \"1.6\")\n    public
fun days(value: Int): Duration = value.toDuration(DurationUnit.DAYS)\n\n    /** Returns a [Duration]
representing the specified [value] number of days. */\n    @SinceKotlin(\"1.5\")\n    @ExperimentalTime\n
@Deprecated(\"Use 'Long.days' extension property from Duration.Companion instead.\",
ReplaceWith(\"value.days\", \"kotlin.time.Duration.Companion.days\"))\n
@DeprecatedSinceKotlin(warningSince = \"1.6\")\n    public fun days(value: Long): Duration =
value.toDuration(DurationUnit.DAYS)\n\n    /**\n    * Returns a [Duration] representing the specified [value]
number of days.\n    *\n    * @throws IllegalArgumentException if the provided `Double` [value] is `NaN`.\n    */
*\n    @SinceKotlin(\"1.5\")\n    @ExperimentalTime\n    @Deprecated(\"Use 'Double.days' extension
property from Duration.Companion instead.\", ReplaceWith(\"value.days\",
\"kotlin.time.Duration.Companion.days\"))\n    @DeprecatedSinceKotlin(warningSince = \"1.6\")\n    public
fun days(value: Double): Duration = value.toDuration(DurationUnit.DAYS)\n\n    /**\n    * Parses a string that
represents a duration and returns the parsed [Duration] value.\n    *\n    * The following formats are
accepted:\n    *\n    * - ISO-8601 Duration format, e.g. `P1DT2H3M4.058S`, see [toIsoString] and
[parseIsoString].\n    * - The format of string returned by the default [Duration.toString] and `toString` in a

```

```

specific unit,\n      * e.g. `10s`, `1h 30m` or `-(1h 30m)`.\n      *\n      * @throws IllegalArgumentException if
the string doesn't represent a duration in any of the supported formats.\n      * @sample
samples.time.Durations.parse\n      *\n      public fun parse(value: String): Duration = try {\n
parseDuration(value, strictIso = false)\n      } catch (e: IllegalArgumentException) {\n      throw
IllegalArgumentException("Invalid duration string format: '$value'.", e)\n      }\n      /**\n      * Parses a
string that represents a duration in ISO-8601 format and returns the parsed [Duration] value.\n      *\n      *
@throws IllegalArgumentException if the string doesn't represent a duration in ISO-8601 format.\n      * @sample
samples.time.Durations.parseIsoString\n      *\n      public fun parseIsoString(value: String): Duration = try {\n
parseDuration(value, strictIso = true)\n      } catch (e: IllegalArgumentException) {\n      throw
IllegalArgumentException("Invalid ISO duration string format: '$value'.", e)\n      }\n      /**\n      * Parses a
string that represents a duration and returns the parsed [Duration] value,\n      * or `null` if the string doesn't
represent a duration in any of the supported formats.\n      *\n      * The following formats are accepted:\n
*\n      * - ISO-8601 Duration format, e.g. `P1DT2H3M4.058S`, see [toISOString] and [parseIsoString].\n      * -
The format of string returned by the default [Duration.toString] and `toString` in a specific unit,\n      * e.g. `10s`,
`1h 30m` or `-(1h 30m)`.\n      * @sample samples.time.Durations.parse\n      *\n      public fun
parseOrNull(value: String): Duration? = try {\n      parseDuration(value, strictIso = false)\n      } catch (e:
IllegalArgumentException) {\n      null\n      }\n      /**\n      * Parses a string that represents a duration in
ISO-8601 format and returns the parsed [Duration] value,\n      * or `null` if the string doesn't represent a duration
in ISO-8601 format.\n      * @sample samples.time.Durations.parseIsoString\n      *\n      public fun
parseIsoStringOrNull(value: String): Duration? = try {\n      parseDuration(value, strictIso = true)\n      } catch
(e: IllegalArgumentException) {\n      null\n      }\n      }\n      // arithmetic operators\n      /** Returns the
negative of this value. *\n      public operator fun unaryMinus(): Duration = durationOf(-value,
unitDiscriminator)\n      /**\n      * Returns a duration whose value is the sum of this and [other] duration values.\n
*\n      * @throws IllegalArgumentException if the operation results in an undefined value for the given arguments,\n
* e.g. when adding infinite durations of different sign.\n      *\n      public operator fun plus(other: Duration):
Duration {\n      when {\n      this.isInfinite() -> {\n      if (other.isFinite() || (this.rawValue xor
other.rawValue >= 0))\n      return this\n      else\n      throw
IllegalArgumentException("Summing infinite durations of different signs yields an undefined result.")\n      }\n
      other.isInfinite() -> return other\n      }\n      return when {\n      this.unitDiscriminator ==
other.unitDiscriminator -> {\n      val result = this.value + other.value // never overflows long, but can
overflow long63\n      when {\n      isInNanos() ->\n
durationOfNanosNormalized(result)\n      else ->\n      durationOfMillisNormalized(result)\n
      }\n      }\n      this.isInMillis() ->\n      addValuesMixedRanges(this.value, other.value)\n
      else ->\n      addValuesMixedRanges(other.value, this.value)\n      }\n      }\n      private fun
addValuesMixedRanges(thisMillis: Long, otherNanos: Long): Duration {\n      val otherMillis =
nanosToMillis(otherNanos)\n      val resultMillis = thisMillis + otherMillis\n      return if (resultMillis in -
MAX_NANOS_IN_MILLIS..MAX_NANOS_IN_MILLIS) {\n      val otherNanoRemainder = otherNanos -
millisToNanos(otherMillis)\n      durationOfNanos(millisToNanos(resultMillis) + otherNanoRemainder)\n      }
      else {\n      durationOfMillis(resultMillis.coerceIn(-MAX_MILLIS, MAX_MILLIS))\n      }\n      }\n      /**\n
* Returns a duration whose value is the difference between this and [other] duration values.\n      *\n      * @throws
IllegalArgumentException if the operation results in an undefined value for the given arguments,\n      * e.g. when
subtracting infinite durations of the same sign.\n      *\n      public operator fun minus(other: Duration): Duration =
this + (-other)\n      /**\n      * Returns a duration whose value is this duration value multiplied by the given [scale]
number.\n      *\n      * @throws IllegalArgumentException if the operation results in an undefined value for the given
arguments,\n      * e.g. when multiplying an infinite duration by zero.\n      *\n      public operator fun times(scale: Int):
Duration {\n      if (isInfinite()) {\n      return when {\n      scale == 0 -> throw
IllegalArgumentException("Multiplying infinite duration by zero yields an undefined result.")\n      scale > 0
-> this\n      else -> -this\n      }\n      }\n      if (scale == 0) return ZERO\n      val value = value\n

```

```

val result = value * scale\n    return if (isInNanos()) {\n        if (value in (MAX_NANOS /
Int.MIN_VALUE)..(-MAX_NANOS / Int.MIN_VALUE)) {\n            // can't overflow nanos range for any
scale\n            durationOfNanos(result)\n        } else {\n            if (result / scale == value) {\n
durationOfNanosNormalized(result)\n            } else {\n                val millis = nanosToMillis(value)\n
val remNanos = value - millisToNanos(millis)\n                val resultMillis = millis * scale\n                val
totalMillis = resultMillis + nanosToMillis(remNanos * scale)\n                if (resultMillis / scale == millis &&
totalMillis xor resultMillis >= 0) {\n                    durationOfMillis(totalMillis.coerceIn(-
MAX_MILLIS..MAX_MILLIS))\n                } else {\n                    if (value.sign * scale.sign > 0) INFINITE
else NEG_INFINITE\n                }\n            }\n        }\n    } else {\n        if (result / scale == value) {\n
durationOfMillis(result.coerceIn(-MAX_MILLIS..MAX_MILLIS))\n        } else {\n            if (value.sign
* scale.sign > 0) INFINITE else NEG_INFINITE\n        }\n    }\n}\n\n/**\n * Returns a duration whose
value is this duration value multiplied by the given [scale] number.\n * The operation may involve rounding
when the result cannot be represented exactly with a [Double] number.\n * @throws
IllegalArgumentException if the operation results in an undefined value for the given arguments,\n * e.g. when
multiplying an infinite duration by zero.\n */\n public operator fun times(scale: Double): Duration {\n    val
intScale = scale.roundToInt()\n    if (intScale.toDouble() == scale) {\n        return times(intScale)\n    }\n
val unit = storageUnit\n    val result = toDouble(unit) * scale\n    return result.toDuration(unit)\n }\n\n
/**\n * Returns a duration whose value is this duration value divided by the given [scale] number.\n * @throws
IllegalArgumentException if the operation results in an undefined value for the given arguments,\n *
e.g. when dividing zero duration by zero.\n */\n public operator fun div(scale: Int): Duration {\n    if (scale ==
0) {\n        return when {\n            isPositive() -> INFINITE\n            isNegative() -> NEG_INFINITE\n
        } else -> throw IllegalArgumentException("Dividing zero duration by zero yields an undefined result.")\n
}\n    }\n    if (isInNanos()) {\n        return durationOfNanos(value / scale)\n    } else {\n        if
(isInfinite())\n            return this * scale.sign\n        val result = value / scale\n        if (result in -
MAX_NANOS_IN_MILLIS..MAX_NANOS_IN_MILLIS) {\n            val rem = millisToNanos(value - (result *
scale)) / scale\n            return durationOfNanos(millisToNanos(result) + rem)\n        }\n        return
durationOfMillis(result)\n    }\n}\n\n/**\n * Returns a duration whose value is this duration value divided
by the given [scale] number.\n * @throws IllegalArgumentException if the operation results in an
undefined value for the given arguments,\n * e.g. when dividing an infinite duration by infinity or zero duration
by zero.\n */\n public operator fun div(scale: Double): Duration {\n    val intScale = scale.roundToInt()\n
if (intScale.toDouble() == scale && intScale != 0) {\n        return div(intScale)\n    }\n    val unit =
storageUnit\n    val result = toDouble(unit) / scale\n    return result.toDuration(unit)\n }\n\n/** Returns a
number that is the ratio of this and [other] duration values. *\n public operator fun div(other: Duration): Double
{\n    val coarserUnit = maxOf(this.storageUnit, other.storageUnit)\n    return this.toDouble(coarserUnit) /
other.toDouble(coarserUnit)\n }\n\n/** Returns true, if the duration value is less than zero. *\n public fun
isNegative(): Boolean = rawValue < 0\n\n/** Returns true, if the duration value is greater than zero. *\n public
fun isPositive(): Boolean = rawValue > 0\n\n/** Returns true, if the duration value is infinite. *\n public fun
isInfinite(): Boolean = rawValue == INFINITE.rawValue || rawValue == NEG_INFINITE.rawValue\n\n/**
Returns true, if the duration value is finite. *\n public fun isFinite(): Boolean = !isInfinite()\n\n/** Returns the
absolute value of this value. The returned value is always non-negative. *\n public val absoluteValue: Duration
get() = if (isNegative()) -this else this\n\n override fun compareTo(other: Duration): Int {\n    val compareBits =
this.rawValue xor other.rawValue\n    if (compareBits < 0 || compareBits.toInt() and 1 == 0) // different signs or
same sign/same range\n        return this.rawValue.compareTo(other.rawValue)\n    // same sign/different
ranges\n    val r = this.unitDiscriminator - other.unitDiscriminator // compare ranges\n    return if (isNegative())
-r else r\n }\n\n // splitting to components\n\n /**\n * Splits this duration into days, hours, minutes,
seconds, and nanoseconds and executes the given [action] with these components.\n * The result of [action] is
returned as the result of this function.\n * - `nanoseconds` represents the whole number of nanoseconds in
this duration, and its absolute value is less than 1_000_000_000;\n * - `seconds` represents the whole number of

```

```

seconds in this duration, and its absolute value is less than 60;\n
 * - `minutes` represents the whole number of minutes in this duration, and its absolute value is less than 60;\n
 * - `hours` represents the whole number of hours in this duration, and its absolute value is less than 24;\n
 * - `days` represents the whole number of days in this duration.\n
 * \n
 * Infinite durations are represented as either [Long.MAX_VALUE] days, or [Long.MIN_VALUE] days (depending on the sign of infinity),\n
 * and zeroes in the lower components.\n
 * \n
 public inline fun <T> toComponents(action: (days: Long, hours: Int, minutes: Int, seconds: Int, nanoseconds: Int) -> T): T {\n
     contract { callsInPlace(action, InvocationKind.EXACTLY_ONCE) }\n
     return action(inWholeDays, hoursComponent, minutesComponent, secondsComponent, nanosecondsComponent)\n
 }\n
 /**\n
  * Splits this duration into hours, minutes, seconds, and nanoseconds and executes the given [action] with these components.\n
  * The result of [action] is returned as the result of this function.\n
  * \n
  * - `nanoseconds` represents the whole number of nanoseconds in this duration, and its absolute value is less than 1_000_000_000;\n
  * - `seconds` represents the whole number of seconds in this duration, and its absolute value is less than 60;\n
  * - `minutes` represents the whole number of minutes in this duration, and its absolute value is less than 60;\n
  * - `hours` represents the whole number of hours in this duration.\n
  * \n
  * Infinite durations are represented as either [Long.MAX_VALUE] hours, or [Long.MIN_VALUE] hours (depending on the sign of infinity),\n
  * and zeroes in the lower components.\n
  * \n
 public inline fun <T> toComponents(action: (hours: Long, minutes: Int, seconds: Int, nanoseconds: Int) -> T): T {\n
     contract { callsInPlace(action, InvocationKind.EXACTLY_ONCE) }\n
     return action(inWholeHours, minutesComponent, secondsComponent, nanosecondsComponent)\n
 }\n
 /**\n
  * Splits this duration into minutes, seconds, and nanoseconds and executes the given [action] with these components.\n
  * The result of [action] is returned as the result of this function.\n
  * \n
  * - `nanoseconds` represents the whole number of nanoseconds in this duration, and its absolute value is less than 1_000_000_000;\n
  * - `seconds` represents the whole number of seconds in this duration, and its absolute value is less than 60;\n
  * - `minutes` represents the whole number of minutes in this duration.\n
  * \n
  * Infinite durations are represented as either [Long.MAX_VALUE] minutes, or [Long.MIN_VALUE] minutes (depending on the sign of infinity),\n
  * and zeroes in the lower components.\n
  * \n
 public inline fun <T> toComponents(action: (minutes: Long, seconds: Int, nanoseconds: Int) -> T): T {\n
     contract { callsInPlace(action, InvocationKind.EXACTLY_ONCE) }\n
     return action(inWholeMinutes, secondsComponent, nanosecondsComponent)\n
 }\n
 /**\n
  * Splits this duration into seconds, and nanoseconds and executes the given [action] with these components.\n
  * The result of [action] is returned as the result of this function.\n
  * \n
  * - `nanoseconds` represents the whole number of nanoseconds in this duration, and its absolute value is less than 1_000_000_000;\n
  * - `seconds` represents the whole number of seconds in this duration.\n
  * \n
  * Infinite durations are represented as either [Long.MAX_VALUE] seconds, or [Long.MIN_VALUE] seconds (depending on the sign of infinity),\n
  * and zero nanoseconds.\n
  * \n
 public inline fun <T> toComponents(action: (seconds: Long, nanoseconds: Int) -> T): T {\n
     contract { callsInPlace(action, InvocationKind.EXACTLY_ONCE) }\n
     return action(inWholeSeconds, nanosecondsComponent)\n
 }\n
 @PublishedApi\n
 internal val hoursComponent: Int\n
 get() = if (isInfinite()) 0 else (inWholeHours % 24).toInt()\n
 @PublishedApi\n
 internal val minutesComponent: Int\n
 get() = if (isInfinite()) 0 else (inWholeMinutes % 60).toInt()\n
 @PublishedApi\n
 internal val secondsComponent: Int\n
 get() = if (isInfinite()) 0 else (inWholeSeconds % 60).toInt()\n
 @PublishedApi\n
 internal val nanosecondsComponent: Int\n
 get() = when {\n
     isInfinite() -> 0\n
     isInMillis() -> millisToNanos(value % 1_000).toInt()\n
     else -> (value % 1_000_000_000).toInt()\n
 }\n
 \n
 // conversion to units\n
 /**\n
  * Returns the value of this duration expressed as a [Double] number of the specified [unit].\n
  * \n
  * The operation may involve rounding when the result cannot be represented exactly with a [Double] number.\n
  * \n
  * An infinite duration value is converted either to [Double.POSITIVE_INFINITY] or [Double.NEGATIVE_INFINITY] depending on its sign.\n
  * \n
 public fun toDouble(unit: DurationUnit): Double {\n
     return when (rawValue) {\n
         INFINITE.rawValue -> Double.POSITIVE_INFINITY\n
         NEG_INFINITE.rawValue -> Double.NEGATIVE_INFINITY\n
     } else -> {\n
         // TODO: whether it's ok to convert to Double before scaling\n
         convertDurationUnit(value.toDouble(), storageUnit, unit)\n
     }\n
 }\n
 /**\n
  * Returns the value

```

```

of this duration expressed as a [Long] number of the specified [unit].\n
 * If the result doesn't fit in the range
of [Long] type, it is coerced into that range:\n
 * - [Long.MIN_VALUE] is returned if it's less than
`Long.MIN_VALUE`,\n
 * - [Long.MAX_VALUE] is returned if it's greater than `Long.MAX_VALUE`.\n
 * An infinite duration value is converted either to [Long.MAX_VALUE] or [Long.MIN_VALUE] depending on
its sign.\n
 * public fun toLong(unit: DurationUnit): Long {\n
return when (rawValue) {\n
INFINITE.rawValue -> Long.MAX_VALUE\n
NEG_INFINITE.rawValue -> Long.MIN_VALUE\n
else -> convertDurationUnit(value, storageUnit, unit)\n
}\n
}\n
/**\n
 * Returns the value of this
duration expressed as an [Int] number of the specified [unit].\n
 * If the result doesn't fit in the range of [Int]
type, it is coerced into that range:\n
 * - [Int.MIN_VALUE] is returned if it's less than `Int.MIN_VALUE`,\n
 * -
[Int.MAX_VALUE] is returned if it's greater than `Int.MAX_VALUE`.\n
 * An infinite duration value is
converted either to [Int.MAX_VALUE] or [Int.MIN_VALUE] depending on its sign.\n
 * public fun
toInt(unit: DurationUnit): Int =\n
toLong(unit).coerceIn(Int.MIN_VALUE.toInt(),
Int.MAX_VALUE.toInt()).toInt()\n
/**\n
 * The value of this duration expressed as a [Double] number of days.
*\n
 * @ExperimentalTime\n
 * @Deprecated("Use inWholeDays property instead or convert toDouble(DAYS) if a
double value is required.", ReplaceWith("toDouble(DurationUnit.DAYS)"))\n
 * public val inDays: Double get() =
toDouble(DurationUnit.DAYS)\n
/**\n
 * The value of this duration expressed as a [Double] number of hours. *\n
 * @ExperimentalTime\n
 * @Deprecated("Use inWholeHours property instead or convert toDouble(HOURS) if a
double value is required.", ReplaceWith("toDouble(DurationUnit.HOURS)"))\n
 * public val inHours: Double
get() = toDouble(DurationUnit.HOURS)\n
/**\n
 * The value of this duration expressed as a [Double] number of
minutes. *\n
 * @ExperimentalTime\n
 * @Deprecated("Use inWholeMinutes property instead or convert
toDouble(MINUTES) if a double value is required.", ReplaceWith("toDouble(DurationUnit.MINUTES)"))\n
 * public val inMinutes: Double get() = toDouble(DurationUnit.MINUTES)\n
/**\n
 * The value of this duration
expressed as a [Double] number of seconds. *\n
 * @ExperimentalTime\n
 * @Deprecated("Use inWholeSeconds
property instead or convert toDouble(SECONDS) if a double value is required.",
ReplaceWith("toDouble(DurationUnit.SECONDS)"))\n
 * public val inSeconds: Double get() =
toDouble(DurationUnit.SECONDS)\n
/**\n
 * The value of this duration expressed as a [Double] number of
milliseconds. *\n
 * @ExperimentalTime\n
 * @Deprecated("Use inWholeMilliseconds property instead or convert
toDouble(MILLISECONDS) if a double value is required.",
ReplaceWith("toDouble(DurationUnit.MILLISECONDS)"))\n
 * public val inMilliseconds: Double get() =
toDouble(DurationUnit.MILLISECONDS)\n
/**\n
 * The value of this duration expressed as a [Double] number of
microseconds. *\n
 * @ExperimentalTime\n
 * @Deprecated("Use inWholeMicroseconds property instead or
convert toDouble(MICROSECONDS) if a double value is required.",
ReplaceWith("toDouble(DurationUnit.MICROSECONDS)"))\n
 * public val inMicroseconds: Double get() =
toDouble(DurationUnit.MICROSECONDS)\n
/**\n
 * The value of this duration expressed as a [Double] number of
nanoseconds. *\n
 * @ExperimentalTime\n
 * @Deprecated("Use inWholeNanoseconds property instead or convert
toDouble(NANOSECONDS) if a double value is required.",
ReplaceWith("toDouble(DurationUnit.NANOSECONDS)"))\n
 * public val inNanoseconds: Double get() =
toDouble(DurationUnit.NANOSECONDS)\n
/**\n
 * The value of this duration expressed as a [Long]
number of days.\n
 * An infinite duration value is converted either to [Long.MAX_VALUE] or
[Long.MIN_VALUE] depending on its sign.\n
 * public val inWholeDays: Long\n
get() =
toLong(DurationUnit.DAYS)\n
/**\n
 * The value of this duration expressed as a [Long] number of hours.\n
 * An infinite duration value is converted either to [Long.MAX_VALUE] or [Long.MIN_VALUE] depending
on its sign.\n
 * public val inWholeHours: Long\n
get() = toLong(DurationUnit.HOURS)\n
/**\n
 * The value of this duration expressed as a [Long] number of minutes.\n
 * An infinite duration value is
converted either to [Long.MAX_VALUE] or [Long.MIN_VALUE] depending on its sign.\n
 * public val
inWholeMinutes: Long\n
get() = toLong(DurationUnit.MINUTES)\n
/**\n
 * The value of this duration
expressed as a [Long] number of seconds.\n
 * An infinite duration value is converted either to
[Long.MAX_VALUE] or [Long.MIN_VALUE] depending on its sign.\n
 * public val inWholeSeconds:

```

```

Long get() = toLong(DurationUnit.SECONDS)
    /**
     * The value of this duration expressed as a
     * [Long] number of milliseconds.
     * An infinite duration value is converted either to [Long.MAX_VALUE]
     * or [Long.MIN_VALUE] depending on its sign.
     */
    public val inWholeMilliseconds: Long get() {
        return if (isInMillis() && isFinite()) value else toLong(DurationUnit.MILLISECONDS)
    }
    /**
     * The value of this duration expressed as a [Long] number of microseconds.
     * If the result doesn't fit in the
     * range of [Long] type, it is coerced into that range:
     * - [Long.MIN_VALUE] is returned if it's less than
     * `Long.MIN_VALUE`,
     * - [Long.MAX_VALUE] is returned if it's greater than `Long.MAX_VALUE`.
     * An infinite duration value is converted either to
     * [Long.MAX_VALUE] or [Long.MIN_VALUE] depending on
     * its sign.
     */
    public val inWholeMicroseconds: Long get() =
        toLong(DurationUnit.MICROSECONDS)
    /**
     * The value of this duration expressed as a [Long] number
     * of nanoseconds.
     * If the result doesn't fit in the range of [Long] type, it is coerced into that range:
     * -
     * [Long.MIN_VALUE] is returned if it's less than `Long.MIN_VALUE`,
     * - [Long.MAX_VALUE] is returned if
     * it's greater than `Long.MAX_VALUE`.
     * An infinite duration value is converted either to
     * [Long.MAX_VALUE] or [Long.MIN_VALUE] depending on its sign.
     */
    public val inWholeNanoseconds: Long get() {
        val value = value
        return when {
            isInNanos() -> value
            value > Long.MAX_VALUE / NANOS_IN_MILLIS -> Long.MAX_VALUE
            value <
            Long.MIN_VALUE / NANOS_IN_MILLIS -> Long.MIN_VALUE
            else -> millisToNanos(value)
        }
    }
    /**
     * Returns the value of this duration expressed as a [Long] number of
     * nanoseconds.
     * If the value doesn't fit in the range of [Long] type, it is coerced into that range, see the
     * conversion [Double.toLong] for details.
     * The range of durations that can be expressed as a `Long`
     * number of nanoseconds is approximately 1292 years.
     */
    @ExperimentalTime
    @Deprecated("Use inWholeNanoseconds property instead.", ReplaceWith("this.inWholeNanoseconds"))
    public fun toLongNanoseconds(): Long = inWholeNanoseconds
    /**
     * Returns the value of this duration expressed as
     * a [Long] number of milliseconds.
     * The value is coerced to the range of [Long] type, if it doesn't fit in
     * that range, see the conversion [Double.toLong] for details.
     * The range of durations that can be expressed
     * as a `Long` number of milliseconds is approximately 1292 million years.
     */
    @ExperimentalTime
    @Deprecated("Use inWholeMilliseconds property instead.", ReplaceWith("this.inWholeMilliseconds"))
    public fun toLongMilliseconds(): Long = inWholeMilliseconds
    /**
     * Returns a string representation of
     * this duration value
     * expressed as a combination of numeric components, each in its own unit.
     * Each
     * component is a number followed by the unit abbreviated name:
     * `d`, `h`, `m`, `s`,
     * `5h`, `1d 12h`, `1h 0m
     * 30.340s`.
     * The last component, usually seconds, can be a number with a fractional part.
     * If the
     * duration is less than a second, it is represented as a single number
     * with one of sub-second units:
     * `ms`
     * (milliseconds), `us` (microseconds), or `ns` (nanoseconds):
     * `140.884ms`, `500us`, `24ns`.
     * A
     * negative duration is prefixed with `-` sign and, if it consists of multiple components, surrounded with parentheses:
     *
     * `-12m` and `-(1h 30m)`.
     * Special cases:
     * - an infinite duration is formatted as `"Infinity"` or
     * `"-Infinity"` without a unit.
     * It's recommended to use [toIsoString] that uses more strict ISO-8601
     * format instead of this `toString` when you want to convert a duration to a string in cases of serialization,
     * interchange, etc.
     */
    @sample samples.time.Durations.toStringDefault
    override fun toString(): String = when (rawValue) {
        0L -> "0s"
        INFINITE.rawValue -> "Infinity"
        NEG_INFINITE.rawValue -> "-Infinity"
        else -> {
            val isNegative = isNegative()
            buildString {
                if (isNegative) append('-')
                absoluteValue.toComponents { days, hours, minutes,
                seconds, nanoseconds ->
                    val hasDays = days != 0
                    val hasHours = hours != 0
                    val hasMinutes = minutes != 0
                    val hasSeconds = seconds != 0 || nanoseconds != 0
                    var
                    components = 0
                    if (hasDays) {
                        append(days).append('d')
                        components++
                    }
                    if (hasHours || (hasDays && (hasMinutes || hasSeconds))) {
                        if (components++ > 0) append(' ')
                        append(hours).append('h')
                    }
                    if
                    (hasMinutes || (hasSeconds && (hasHours || hasDays))) {
                        if (components++ > 0) append(' ')
                        append(minutes).append('m')
                    }
                    if (hasSeconds) {
                        if
                    }
                }
            }
        }
    }
}

```

```

(components++ > 0) append(' ')\n                when {\n                seconds != 0 || hasDays || hasHours ||
hasMinutes ->\n                appendFractional(seconds, nanoseconds, 9, \"s\", isoZeroes = false)\n
                nanoseconds >= 1_000_000 ->\n                appendFractional(nanoseconds / 1_000_000, nanoseconds
% 1_000_000, 6, \"ms\", isoZeroes = false)\n                nanoseconds >= 1_000 ->\n
appendFractional(nanoseconds / 1_000, nanoseconds % 1_000, 3, \"us\", isoZeroes = false)\n                else -
>\n                append(nanoseconds).append(\"ns\")\n                }\n                }\n                if
(isNegative && components > 1) insert(1, (').append('))\n                }\n                }\n                }\n                private fun
StringBuilder.appendFractional(whole: Int, fractional: Int, fractionalSize: Int, unit: String, isoZeroes: Boolean) {\n
                append(whole)\n                if (fractional != 0) {\n                append('.')\n                val fracString =
fractional.toString().padStart(fractionalSize, '0')\n                val nonZeroDigits = fracString.indexOfLast { it != '0' } +
1\n                when {\n                !isoZeroes && nonZeroDigits < 3 -> appendRange(fracString, 0, nonZeroDigits)\n
                else -> appendRange(fracString, 0, ((nonZeroDigits + 2) / 3) * 3)\n                }\n                }\n                append(unit)\n
}\n\n /**\n * Returns a string representation of this duration value expressed in the given [unit]\n * and
formatted with the specified [decimals] number of digits after decimal point.\n * \n * Special cases:\n * - an
infinite duration is formatted as `\"Infinity\"` or `\"-Infinity\"` without a unit.\n * \n * @param decimals the
number of digits after decimal point to show. The value must be non-negative.\n * \n * No more than 12 decimals will
be shown, even if a larger number is requested.\n * \n * @return the value of duration in the specified [unit]
followed by that unit abbreviated name: `d`, `h`, `m`, `s`, `ms`, `us`, or `ns`.\n * \n * @throws
IllegalArgumentExcepion if [decimals] is less than zero.\n * \n * @sample
samples.time.Durations.toStringDecimals\n * \n public fun toString(unit: DurationUnit, decimals: Int = 0):
String {\n     require(decimals >= 0) { \"decimals must be not negative, but was $decimals\" }\n     val number =
toDouble(unit)\n     if (number.isInfinite()) return number.toString()\n     return formatToExactDecimals(number,
decimals.coerceAtMost(12)) + unit.shortName()\n }\n\n /**\n * Returns an ISO-8601 based string
representation of this duration.\n * \n * The returned value is presented in the format `PTm>s.fS`, where `h`,
`m`, `s` are the integer components of this duration (see [toComponents])\n * and `f` is a fractional part of second.
Depending on the roundness of the value the fractional part can be formatted with either\n * 0, 3, 6, or 9 decimal
digits.\n * \n * The infinite duration is represented as `\"PT9999999999999999H\"` which is larger than any
possible finite duration in Kotlin.\n * \n * Negative durations are indicated with the sign `-` in the beginning of
the returned string, for example, `\"-PT5M30S\"`.\n * \n * @sample samples.time.Durations.toIsoString\n
*\n public fun toIsoString(): String = buildString {\n     if (isNegative()) append('-')\n     append(\"PT\")\n
this@Duration.absoluteValue.toComponents { hours, minutes, seconds, nanoseconds ->\n
@Suppress(\"NAME_SHADOWING\")\n         var hours = hours\n         if (isInfinite()) {\n         // use large
enough value instead of Long.MAX_VALUE\n         hours = 9_999_999_999_999\n         }\n         val
hasHours = hours != 0L\n         val hasSeconds = seconds != 0 || nanoseconds != 0\n         val hasMinutes =
minutes != 0 || (hasSeconds && hasHours)\n         if (hasHours) {\n         append(hours).append('H')\n
}\n         if (hasMinutes) {\n         append(minutes).append('M')\n         }\n         if (hasSeconds ||
(!hasHours && !hasMinutes)) {\n         appendFractional(seconds, nanoseconds, 9, \"S\", isoZeroes = true)\n
}\n         }\n         }\n         }\n         }\n\n // constructing from number of units\n // extension functions\n /** Returns a [Duration]
equal to this [Int] number of the specified [unit].
*\n @SinceKotlin(\"1.6\")\n @WasExperimental(ExperimentalTime::class)\n public fun Int.toDuration(unit:
DurationUnit): Duration {\n     return if (unit <= DurationUnit.SECONDS) {\n
durationOfNanos(convertDurationUnitOverflow(this.toLong(), unit, DurationUnit.NANOSECONDS))\n     } else {\n
toLong().toDuration(unit)\n     }\n }\n /** Returns a [Duration] equal to this [Long] number of the specified [unit].
*\n @SinceKotlin(\"1.6\")\n @WasExperimental(ExperimentalTime::class)\n public fun Long.toDuration(unit:
DurationUnit): Duration {\n     val maxNsInUnit = convertDurationUnitOverflow(MAX_NANOS,
DurationUnit.NANOSECONDS, unit)\n     if (this in -maxNsInUnit..maxNsInUnit) {\n     return
durationOfNanos(convertDurationUnitOverflow(this, unit, DurationUnit.NANOSECONDS))\n     } else {\n     val
millis = convertDurationUnit(this, unit, DurationUnit.MILLISECONDS)\n     return

```

```

durationOfMillis(millis.coerceIn(-MAX_MILLIS, MAX_MILLIS))\n    }\n}\n\n/**\n * Returns a [Duration] equal
to this [Double] number of the specified [unit].\n *\n * Depending on its magnitude, the value is rounded to an
integer number of nanoseconds or milliseconds.\n *\n * @throws IllegalArgumentException if this `Double` value is
`NaN`.\n */\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalTime::class)\npublic fun
Double.toDuration(unit: DurationUnit): Duration {\n    val valueInNs = convertDurationUnit(this, unit,
DurationUnit.NANOSECONDS)\n    require(!valueInNs.isNaN()) { "Duration value cannot be NaN." }\n    val
nanos = valueInNs.roundToLong()\n    return if (nanos in -MAX_NANOS..MAX_NANOS) {\n
durationOfNanos(nanos)\n    } else {\n        val millis = convertDurationUnit(this, unit,
DurationUnit.MILLISECONDS).roundToLong()\n        durationOfMillisNormalized(millis)\n    }\n}\n\n//
constructing from number of units\n// deprecated extension properties\n\n/** Returns a [Duration] equal to this [Int]
number of nanoseconds. *\n */\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Int.nanoseconds'
extension property from Duration.Companion instead.", ReplaceWith("this.nanoseconds",
"\"kotlin.time.Duration.Companion.nanoseconds\""))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Int.nanoseconds get() = toDuration(DurationUnit.NANOSECONDS)\n\n/** Returns a [Duration] equal to this
[Long] number of nanoseconds. *\n */\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use
'Long.nanoseconds' extension property from Duration.Companion instead.", ReplaceWith("this.nanoseconds",
"\"kotlin.time.Duration.Companion.nanoseconds\""))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Long.nanoseconds get() = toDuration(DurationUnit.NANOSECONDS)\n\n/**\n *\n * Returns a [Duration] equal to this
[Double] number of nanoseconds.\n *\n * @throws IllegalArgumentException if this [Double] value is `NaN`.\n
*/\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Double.nanoseconds' extension property
from Duration.Companion instead.", ReplaceWith("this.nanoseconds",
"\"kotlin.time.Duration.Companion.nanoseconds\""))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Double.nanoseconds get() = toDuration(DurationUnit.NANOSECONDS)\n\n/** Returns a [Duration] equal to
this [Int] number of microseconds. *\n */\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use
'Int.microseconds' extension property from Duration.Companion instead.", ReplaceWith("this.microseconds",
"\"kotlin.time.Duration.Companion.microseconds\""))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Int.microseconds get() = toDuration(DurationUnit.MICROSECONDS)\n\n/** Returns a [Duration] equal to this
[Long] number of microseconds. *\n */\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use
'Long.microseconds' extension property from Duration.Companion instead.", ReplaceWith("this.microseconds",
"\"kotlin.time.Duration.Companion.microseconds\""))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Long.microseconds get() = toDuration(DurationUnit.MICROSECONDS)\n\n/**\n *\n * Returns a [Duration] equal to
this [Double] number of microseconds.\n *\n * @throws IllegalArgumentException if this [Double] value is
`NaN`.\n */\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Double.microseconds' extension
property from Duration.Companion instead.", ReplaceWith("this.microseconds",
"\"kotlin.time.Duration.Companion.microseconds\""))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Double.microseconds get() = toDuration(DurationUnit.MICROSECONDS)\n\n/** Returns a [Duration] equal to
this [Int] number of milliseconds. *\n */\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use
'Int.milliseconds' extension property from Duration.Companion instead.", ReplaceWith("this.milliseconds",
"\"kotlin.time.Duration.Companion.milliseconds\""))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Int.milliseconds get() = toDuration(DurationUnit.MILLISECONDS)\n\n/** Returns a [Duration] equal to this
[Long] number of milliseconds. *\n */\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use
'Long.milliseconds' extension property from Duration.Companion instead.", ReplaceWith("this.milliseconds",
"\"kotlin.time.Duration.Companion.milliseconds\""))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Long.milliseconds get() = toDuration(DurationUnit.MILLISECONDS)\n\n/**\n *\n * Returns a [Duration] equal to this
[Double] number of milliseconds.\n *\n * @throws IllegalArgumentException if this [Double] value is `NaN`.\n
*/\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Double.milliseconds' extension property
from Duration.Companion instead.", ReplaceWith("this.milliseconds",
"\"kotlin.time.Duration.Companion.milliseconds\""))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val

```

```

Double.milliseconds get() = toDuration(DurationUnit.MILLISECONDS)\n\n/** Returns a [Duration] equal to this
[Int] number of seconds. *\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Int.seconds'
extension property from Duration.Companion instead.", ReplaceWith("this.seconds",
"kotlin.time.Duration.Companion.seconds"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Int.seconds get() = toDuration(DurationUnit.SECONDS)\n\n/** Returns a [Duration] equal to this [Long] number of
seconds. *\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Long.seconds' extension property
from Duration.Companion instead.", ReplaceWith("this.seconds",
"kotlin.time.Duration.Companion.seconds"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Long.seconds get() = toDuration(DurationUnit.SECONDS)\n\n/**\n * Returns a [Duration] equal to this [Double]
number of seconds.\n *\n * @throws IllegalArgumentException if this [Double] value is `NaN`.\n
*\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Double.seconds' extension property from
Duration.Companion instead.", ReplaceWith("this.seconds",
"kotlin.time.Duration.Companion.seconds"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Double.seconds get() = toDuration(DurationUnit.SECONDS)\n\n/** Returns a [Duration] equal to this [Int]
number of minutes. *\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Int.minutes' extension
property from Duration.Companion instead.", ReplaceWith("this.minutes",
"kotlin.time.Duration.Companion.minutes"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Int.minutes get() = toDuration(DurationUnit.MINUTES)\n\n/** Returns a [Duration] equal to this [Long] number of
minutes. *\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Long.minutes' extension property
from Duration.Companion instead.", ReplaceWith("this.minutes",
"kotlin.time.Duration.Companion.minutes"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Long.minutes get() = toDuration(DurationUnit.MINUTES)\n\n/**\n * Returns a [Duration] equal to this [Double]
number of minutes.\n *\n * @throws IllegalArgumentException if this [Double] value is `NaN`.\n
*\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Double.minutes' extension property from
Duration.Companion instead.", ReplaceWith("this.minutes",
"kotlin.time.Duration.Companion.minutes"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Double.minutes get() = toDuration(DurationUnit.MINUTES)\n\n/** Returns a [Duration] equal to this [Int]
number of hours. *\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Int.hours' extension
property from Duration.Companion instead.", ReplaceWith("this.hours",
"kotlin.time.Duration.Companion.hours"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Int.hours get() = toDuration(DurationUnit.HOURS)\n\n/** Returns a [Duration] equal to this [Long] number of
hours. *\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Long.hours' extension property from
Duration.Companion instead.", ReplaceWith("this.hours",
"kotlin.time.Duration.Companion.hours"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Long.hours get() = toDuration(DurationUnit.HOURS)\n\n/**\n * Returns a [Duration] equal to this [Double]
number of hours.\n *\n * @throws IllegalArgumentException if this [Double] value is `NaN`.\n
*\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Double.hours' extension property from
Duration.Companion instead.", ReplaceWith("this.hours",
"kotlin.time.Duration.Companion.hours"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Double.hours get() = toDuration(DurationUnit.HOURS)\n\n/** Returns a [Duration] equal to this [Int] number of
days. *\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Int.days' extension property from
Duration.Companion instead.", ReplaceWith("this.days",
"kotlin.time.Duration.Companion.days"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val Int.days
get() = toDuration(DurationUnit.DAYS)\n\n/** Returns a [Duration] equal to this [Long] number of days.
*\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Long.days' extension property from
Duration.Companion instead.", ReplaceWith("this.days",
"kotlin.time.Duration.Companion.days"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Long.days get() = toDuration(DurationUnit.DAYS)\n\n/**\n * Returns a [Duration] equal to this [Double] number

```

```

of days.\n *\n * @throws IllegalArgumentException if this [Double] value is `NaN`.\n
*\n@SinceKotlin("1.3")\n@ExperimentalTime\n@Deprecated("Use 'Double.days' extension property from
Duration.Companion instead.", ReplaceWith("this.days"),
"kotlin.time.Duration.Companion.days")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic val
Double.days get() = toDuration(DurationUnit.DAYS)\n\n/** Returns a duration whose value is the specified
[duration] value multiplied by this number.
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalTime::class)\n@kotlin.internal.InlineOnly\npublic
inline operator fun Int.times(duration: Duration): Duration = duration * this\n\n/** Returns a duration whose
value is the specified [duration] value multiplied by this number.\n *\n * The operation may involve rounding when
the result cannot be represented exactly with a [Double] number.\n *\n * @throws IllegalArgumentException if the
operation results in a `NaN` value.\n
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalTime::class)\n@kotlin.internal.InlineOnly\npublic
inline operator fun Double.times(duration: Duration): Duration = duration * this\n\n\nprivate fun
parseDuration(value: String, strictIso: Boolean): Duration {\n    var length = value.length\n    if (length == 0) throw
IllegalArgumentException("The string is empty")\n    var index = 0\n    var result = Duration.ZERO\n    val
infinityString = "Infinity"\n    when (value[index]) {\n        '+', '-' -> index++\n    }\n    val hasSign = index > 0\n    val isNegative = hasSign && value.startsWith('-')\n    when {\n        length <= index ->\n            throw
IllegalArgumentException("No components")\n        value[index] == 'P' -> {\n            if (++index == length) throw
IllegalArgumentException()\n            val nonDigitSymbols = "+-." \n            var isTimeComponent = false\n            var prevUnit: DurationUnit? = null\n            while (index < length) {\n                if (value[index] == 'T') {\n                    if (isTimeComponent || ++index == length) throw
IllegalArgumentException()\n                    isTimeComponent =
true\n                    continue\n                }\n                val component = value.substringWhile(index) { it in '0'..'9' || it in
nonDigitSymbols }\n                if (component.isEmpty()) throw
IllegalArgumentException()\n                index +=
component.length\n                val unitChar = value.getOrElse(index) { throw
IllegalArgumentException("Missing unit for value $component") }\n                index++\n                val unit = durationUnitByIsoChar(unitChar,
isTimeComponent)\n                if (prevUnit != null && prevUnit <= unit) throw
IllegalArgumentException("Unexpected order of duration components")\n                prevUnit = unit\n                val
dotIndex = component.indexOf('.')\n                if (unit == DurationUnit.SECONDS && dotIndex > 0) {\n                    val whole = component.substring(0, dotIndex)\n                    result +=
parseOverLongIsoComponent(whole).toDuration(unit)\n                    result +=
component.substring(dotIndex).toDouble().toDuration(unit)\n                } else {\n                    result +=
parseOverLongIsoComponent(component).toDuration(unit)\n                }\n            }\n            strictIso ->\n                throw
IllegalArgumentException() \n            value.regionMatches(index, infinityString, 0, length = maxOf(length -
index, infinityString.length), ignoreCase = true) -> {\n                result = Duration.INFINITE\n            } \n            else -> {\n                // parse default string format\n                var prevUnit: DurationUnit? = null\n                var afterFirst = false\n                var allowSpaces = !hasSign\n                if (hasSign && value[index] == '(' && value.last() == ')') {\n                    allowSpaces = true\n                    if (++index == --length) throw
IllegalArgumentException("No components")\n                }\n                while (index < length) {\n                    if (afterFirst && allowSpaces) {\n                        index =
value.skipWhile(index) { it == ' ' }\n                    }\n                    afterFirst = true\n                    val component =
value.substringWhile(index) { it in '0'..'9' || it == '.' }\n                    if (component.isEmpty()) throw
IllegalArgumentException()\n                    index += component.length\n                    val unitName =
value.substringWhile(index) { it in 'a'..'z' }\n                    index += unitName.length\n                    val unit =
durationUnitByShortName(unitName)\n                    if (prevUnit != null && prevUnit <= unit) throw
IllegalArgumentException("Unexpected order of duration components")\n                    prevUnit = unit\n                    val
dotIndex = component.indexOf('.')\n                    if (dotIndex > 0) {\n                        val whole = component.substring(0,
dotIndex)\n                        result += whole.toLong().toDuration(unit)\n                        result +=
component.substring(dotIndex).toDouble().toDuration(unit)\n                    } \n                    if (index < length) throw
IllegalArgumentException("Fractional component must be last")\n                } else {\n                    result +=

```

```

component.toLong().toDuration(unit)\n          }\n          }\n          }\n          } return if (isNegative) -result else
result\n}\n\nprivate fun parseOverLongIsoComponent(value: String): Long {\n    val length = value.length\n    var
startIndex = 0\n    if (length > 0 && value[0] in '"+-') startIndex++\n    if ((length - startIndex) > 16 &&
(startIndex..value.lastIndex).all { value[it] in '0'..'9' }) {\n        // all chars are digits, but more than
ceiling(log10(MAX_MILLIS / 1000)) of them\n        return if (value[0] == '-') Long.MIN_VALUE else
Long.MAX_VALUE\n    }\n    // TODO: replace with just toLong after min JDK becomes 8\n    return if
(value.startsWith '"+-')) value.drop(1).toLong() else value.toLong()\n}\n\nprivate inline fun
String.substringWhile(startIndex: Int, predicate: (Char) -> Boolean): String =\n    substring(startIndex,
skipWhile(startIndex, predicate))\n\nprivate inline fun String.skipWhile(startIndex: Int, predicate: (Char) ->
Boolean): Int {\n    var i = startIndex\n    while (i < length && predicate(this[i])) i++\n    return i\n}\n\n\n\n\n\n//
The ranges are chosen so that they are:\n// - symmetric relative to zero: this greatly simplifies operations with sign,
e.g. unaryMinus and minus.\n// - non-overlapping, but adjacent: the first value that doesn't fit in nanos range, can be
exactly represented in millis.\n\ninternal const val NANOS_IN_MILLIS = 1_000_000\n// maximum number
duration can store in nanosecond range\ninternal const val MAX_NANOS = Long.MAX_VALUE / 2 /
NANOS_IN_MILLIS * NANOS_IN_MILLIS - 1 // ends in ...999_999\n// maximum number duration can store in
millisecond range, also encodes an infinite value\ninternal const val MAX_MILLIS = Long.MAX_VALUE / 2\n//
MAX_NANOS expressed in milliseconds\nprivate const val MAX_NANOS_IN_MILLIS = MAX_NANOS /
NANOS_IN_MILLIS\n\nprivate fun nanosToMillis(nanos: Long): Long = nanos / NANOS_IN_MILLIS\nprivate
fun millisToNanos(millis: Long): Long = millis * NANOS_IN_MILLIS\n\nprivate fun
durationOfNanos(normalNanos: Long) = Duration(normalNanos shl 1)\nprivate fun durationOfMillis(normalMillis:
Long) = Duration((normalMillis shl 1) + 1)\nprivate fun durationOf(normalValue: Long, unitDiscriminator: Int) =
Duration((normalValue shl 1) + unitDiscriminator)\nprivate fun durationOfNanosNormalized(nanos: Long) =\n    if
(nanos in -MAX_NANOS..MAX_NANOS) {\n        durationOfNanos(nanos)\n    } else {\n
durationOfMillis(nanosToMillis(nanos))\n    }\n\nprivate fun durationOfMillisNormalized(millis: Long) =\n    if
(millis in -MAX_NANOS_IN_MILLIS..MAX_NANOS_IN_MILLIS) {\n
durationOfNanos(millisToNanos(millis))\n    } else {\n        durationOfMillis(millis.coerceIn(-MAX_MILLIS,
MAX_MILLIS))\n    }\n\ninternal expect val durationAssertionsEnabled: Boolean\n\ninternal expect fun
formatToExactDecimals(value: Double, decimals: Int): String\n\ninternal expect fun formatUpToDecimals(value:
Double, decimals: Int): String", "/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n@file:kotlin.jvm.JvmName("UnsignedKt")\npackage
kotlin\n\n@PublishedApi\ninternal fun uintCompare(v1: Int, v2: Int): Int = (v1 xor
Int.MIN_VALUE).compareTo(v2 xor Int.MIN_VALUE)\n\n@PublishedApi\ninternal fun ulongCompare(v1: Long,
v2: Long): Int = (v1 xor Long.MIN_VALUE).compareTo(v2 xor Long.MIN_VALUE)\n\n@PublishedApi\ninternal
fun uintDivide(v1: UInt, v2: UInt): UInt = (v1.toLong() / v2.toLong()).toUInt()\n\n@PublishedApi\ninternal fun
uintRemainder(v1: UInt, v2: UInt): UInt = (v1.toLong() % v2.toLong()).toUInt()\n\n// Division and remainder are
based on Guava's UnsignedLongs implementation\n// Copyright 2011 The Guava
Authors\n\n@PublishedApi\ninternal fun ulongDivide(v1: ULong, v2: ULong): ULong {\n    val dividend =
v1.toLong()\n    val divisor = v2.toLong()\n    if (divisor < 0) { // i.e., divisor >= 2^63:\n        return if (v1 < v2)
ULong(0) else ULong(1)\n    }\n\n    // Optimization - use signed division if both dividend and divisor < 2^63\n    if
(dividend >= 0) {\n        return ULong(dividend / divisor)\n    }\n\n    // Otherwise, approximate the quotient, check,
and correct if necessary.\n    val quotient = ((dividend ushr 1) / divisor) shl 1\n    val rem = dividend - quotient *
divisor\n    return ULong(quotient + if (ULong(rem) >= ULong(divisor)) 1 else 0)\n}\n\n@PublishedApi\ninternal
fun ulongRemainder(v1: ULong, v2: ULong): ULong {\n    val dividend = v1.toLong()\n    val divisor =
v2.toLong()\n    if (divisor < 0) { // i.e., divisor >= 2^63:\n        return if (v1 < v2) {\n            v1 // dividend <
divisor\n        } else {\n            v1 - v2 // dividend >= divisor\n        }\n    }\n\n    // Optimization - use signed
modulus if both dividend and divisor < 2^63\n    if (dividend >= 0) {\n        return ULong(dividend % divisor)\n
    }\n\n    // Otherwise, approximate the quotient, check, and correct if necessary.\n    val quotient = ((dividend ushr 1)

```

```

/divisor) shl 1\n    val rem = dividend - quotient * divisor\n    return ULong(rem - if (ULong(rem) >=
ULong(divisor)) divisor else 0)\n}\n\n@PublishedApi\ninternal fun doubleToUInt(v: Double): UInt = when {\n
v.isNaN() -> 0u\n    v <= UInt.MIN_VALUE.toDouble() -> UInt.MIN_VALUE\n    v >=
UInt.MAX_VALUE.toDouble() -> UInt.MAX_VALUE\n    v <= Int.MAX_VALUE -> v.toInt().toUInt()\n    else -
> (v - Int.MAX_VALUE).toInt().toUInt() + Int.MAX_VALUE.toUInt()    // Int.MAX_VALUE < v <
UInt.MAX_VALUE\n}\n\n@PublishedApi\ninternal fun doubleToULong(v: Double): ULong = when {\n
v.isNaN() -> 0u\n    v <= ULong.MIN_VALUE.toDouble() -> ULong.MIN_VALUE\n    v >=
ULong.MAX_VALUE.toDouble() -> ULong.MAX_VALUE\n    v < Long.MAX_VALUE ->
v.toLong().toULong()\n    // Real values from Long.MAX_VALUE to (Long.MAX_VALUE + 1) are not
representable in Double, so don't handle them.\n    else -> (v - 9223372036854775808.0).toLong().toULong() +
9223372036854775808uL    // Long.MAX_VALUE + 1 < v <
ULong.MAX_VALUE\n}\n\n\n@PublishedApi\ninternal fun uintToDouble(v: Int): Double = (v and
Int.MAX_VALUE).toDouble() + (v ushr 31 shl 30).toDouble() * 2\n\n@PublishedApi\ninternal fun
ulongToDouble(v: Long): Double = (v ushr 11).toDouble() * 2048 + (v and 2047)\n\n\ninternal fun
ulongToString(v: Long): String = ulongToString(v, 10)\n\n\ninternal fun ulongToString(v: Long, base: Int): String {\n
    if (v >= 0) return v.toString(base)\n    var quotient = ((v ushr 1) / base) shl 1\n    var rem = v - quotient * base\n
    if (rem >= base) {\n        rem -= base\n        quotient += 1\n    }\n    return quotient.toString(base) +
rem.toString(base)\n}\n\n", "/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n
*\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("\CollectionsKt")\n\npackage
kotlin.collections\n\n/**\n * Given an [iterator] function constructs an [Iterable] instance that returns values through
the [Iterator]\n * provided by that function.\n * @sample samples.collections.Iterables.Building.iterable\n
*\n@kotlin.internal.InlineOnly\npublic inline fun <T> Iterable(crossinline iterator: () -> Iterator<T>): Iterable<T>
= object : Iterable<T> {\n    override fun iterator(): Iterator<T> = iterator()\n}\n\n/**\n * A wrapper over another
[Iterable] (or any other object that can produce an [Iterator]) that returns\n * an indexing iterator.\n *\ninternal class
IndexingIterable<out T>(private val iteratorFactory: () -> Iterator<T>) : Iterable<IndexedValue<T>> {\n    override
fun iterator(): Iterator<IndexedValue<T>> = IndexingIterator(iteratorFactory())\n}\n\n\n/**\n * Returns the size of
this iterable if it is known, or `null` otherwise.\n *\n@PublishedApi\ninternal fun <T>
Iterable<T>.collectionSizeOrNull(): Int? = if (this is Collection<*>) this.size else null\n\n\n/**\n * Returns the size of
this iterable if it is known, or the specified [default] value otherwise.\n *\n@PublishedApi\ninternal fun <T>
Iterable<T>.collectionSizeOrDefault(default: Int): Int = if (this is Collection<*>) this.size else default\n\n\n\n/**\n *
Returns a single list of all elements from all collections in the given collection.\n * @sample
samples.collections.Iterables.Operations.flattenIterable\n *\npublic fun <T> Iterable<Iterable<T>>.flatten():
List<T> {\n    val result = ArrayList<T>()\n    for (element in this) {\n        result.addAll(element)\n    }\n    return
result\n}\n\n\n/**\n * Returns a pair of lists, where\n * *first* list is built from the first values of each pair from this
collection,\n * *second* list is built from the second values of each pair from this collection.\n * @sample
samples.collections.Iterables.Operations.unzipIterable\n *\npublic fun <T, R> Iterable<Pair<T, R>>.unzip():
Pair<List<T>, List<R>> {\n    val expectedSize = collectionSizeOrDefault(10)\n    val listT =
ArrayList<T>(expectedSize)\n    val listR = ArrayList<R>(expectedSize)\n    for (pair in this) {\n
listT.add(pair.first)\n        listR.add(pair.second)\n    }\n    return listT to listR\n}\n\n", "/*\n * Copyright 2010-2020
JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("\SequencesKt")\n\npackage
kotlin.sequences\n\nimport kotlin.random.Random\n\n\n/**\n * Given an [iterator] function constructs a [Sequence]
that returns values through the [Iterator]\n * provided by that function.\n * The values are evaluated lazily, and the
sequence is potentially infinite.\n *\n *\n * @sample samples.collections.Sequences.Building.sequenceFromIterator\n
*\n@kotlin.internal.InlineOnly\npublic inline fun <T> Sequence(crossinline iterator: () -> Iterator<T>):

```

```

Sequence<T> = object : Sequence<T> {
    override fun iterator(): Iterator<T> = iterator()
}

// Creates a sequence that returns all elements from this iterator. The sequence is constrained to be iterated only once.
@sample samples.collections.Sequences.Building.sequenceFromIterator

// public fun <T>
// Iterator<T>.asSequence(): Sequence<T> = Sequence { this }.constrainOnce()
// Creates a sequence that returns the specified values.
@sample samples.collections.Sequences.Building.sequenceOfValues

// public fun <T> sequenceOf(vararg elements: T): Sequence<T> = if (elements.isEmpty()) emptySequence() else
// elements.asSequence()
// Returns an empty sequence.
// public fun <T> emptySequence():
// Sequence<T> = EmptySequence

// private object EmptySequence : Sequence<Nothing>,
// DropTakeSequence<Nothing> {
//     override fun iterator(): Iterator<Nothing> = EmptyIterator
//     override fun
//     drop(n: Int) = EmptySequence
//     override fun take(n: Int) = EmptySequence
// }
// Returns this sequence if it's not `null` and the empty sequence otherwise.
@sample
samples.collections.Sequences.Usage.sequenceOrEmpty

// public fun <T> Sequence<T>?.orEmpty():
// Sequence<T> = this ?: emptySequence()
// Returns a sequence that iterates through the elements either of this sequence or, if this sequence turns out to be empty, of the sequence returned by [defaultValue] function.
@sample samples.collections.Sequences.Usage.sequenceIfEmpty

// public fun
// <T> Sequence<T>.ifEmpty(defaultValue: () -> Sequence<T>): Sequence<T> = sequence {
//     val iterator =
//     this.ifEmpty().iterator()
//     if (iterator.hasNext()) {
//         yieldAll(iterator)
//     } else {
//         yieldAll(defaultValue())
//     }
// }
// Returns a sequence of all elements from all sequences in this sequence.
// The operation is _intermediate_ and _stateless_.
@sample
samples.collections.Sequences.Transformations.flattenSequenceOfSequences

// public fun <T>
// Sequence<Sequence<T>>.flatten(): Sequence<T> = flatten { it.iterator() }
// Returns a sequence of all elements from all iterables in this sequence.
// The operation is _intermediate_ and _stateless_.
@sample samples.collections.Sequences.Transformations.flattenSequenceOfLists

// public fun
// <T> Sequence<Iterable<T>>.flatten():
// Sequence<T> = flatten { it.iterator() }

// private fun <T, R> Sequence<T>.flatten(iterator: (T) -> Iterator<R>):
// Sequence<R> {
//     if (this is TransformingSequence<*, *>) {
//         return (this as TransformingSequence<*,
//         T>).flatten(iterator)
//     }
//     return FlatteningSequence(this, { it }, iterator)
// }
// Returns a pair of lists, where
// *first* list is built from the first values of each pair from this sequence,
// *second* list is built from the second values of each pair from this sequence.
// The operation is _terminal_.
@sample
samples.collections.Sequences.Transformations.unzip

// public fun <T, R> Sequence<Pair<T, R>>.unzip():
// Pair<List<T>, List<R>> {
//     val listT = ArrayList<T>()
//     val listR = ArrayList<R>()
//     for (pair in this) {
//         listT.add(pair.first)
//         listR.add(pair.second)
//     }
//     return listT to listR
// }
// Returns a sequence that yields elements of this sequence randomly shuffled.
// Note that every iteration of the sequence returns elements in a different order.
// The operation is _intermediate_ and _stateful_.
@sample
samples.collections.Sequences.Transformations.shuffle

// public fun <T> Sequence<T>.shuffled(): Sequence<T> = shuffled(Random)
// Returns a sequence that yields elements of this sequence randomly shuffled
// using the specified [random] instance as the source of randomness.
// Note that every iteration of the sequence returns elements in a different order.
// The operation is _intermediate_ and _stateful_.
@sample
samples.collections.Sequences.Transformations.shuffleWithPredicate

// public fun <T>
// Sequence<T>.shuffled(random: Random): Sequence<T> = sequence<T> {
//     val buffer = toMutableList()
//     while (buffer.isNotEmpty()) {
//         val j = random.nextInt(buffer.size)
//         val last = buffer.removeLast()
//         val value = if (j < buffer.size) buffer.set(j, last) else last
//         yield(value)
//     }
// }
// A sequence that returns the values from the underlying [sequence] that either match or do not match
// the specified [predicate].
// @param sendWhen If `true`, values for which the predicate returns `true` are returned. Otherwise,
// values for which the predicate returns `false` are returned.
// internal class FilteringSequence<T> {
//     private val
//     sequence: Sequence<T>,
//     private val sendWhen: Boolean = true,
//     private val predicate: (T) -> Boolean
// }

// Sequence<T> {
//     override fun iterator(): Iterator<T> = object : Iterator<T> {
//         val iterator =
//         sequence.iterator()
//         var nextState: Int = -1 // -1 for unknown, 0 for done, 1 for continue
//         var nextItem: T?

```

```

= null\n\n    private fun calcNext() {\n        while (iterator.hasNext()) {\n            val item = iterator.next()\n            if (predicate(item) == sendWhen) {\n                nextItem = item\n                nextState = 1\n            }\n        }\n        return\n    }\n\n    override fun next(): T {\n        if (nextState == -1)\n            calcNext()\n        if (nextState == 0)\n            throw NoSuchElementException()\n        val result = nextItem\n        nextItem = null\n        nextState = -1\n        @Suppress("UNCHECKED_CAST")\n        return result as T\n    }\n\n    override fun hasNext(): Boolean {\n        if (nextState == -1)\n            calcNext()\n        return nextState == 1\n    }\n}\n\n/**\n * A sequence which returns the results of applying the given [transformer] function to the values\n * in the underlying [sequence].\n */\ninternal class TransformingSequence<T, R>\nconstructor(private val sequence: Sequence<T>, private val transformer: (T) -> R) : Sequence<R> {\n    override fun iterator(): Iterator<R> = object : Iterator<R> {\n        val iterator = sequence.iterator()\n        override fun next(): R {\n            return transformer(iterator.next())\n        }\n    }\n\n    override fun hasNext(): Boolean {\n        return iterator.hasNext()\n    }\n}\n\ninternal fun <E> flatten(iterator: (R) -> Iterator<E>): Sequence<E> {\n    return FlatteningSequence<T, R, E>(sequence, transformer, iterator)\n}\n\n/**\n * A sequence which returns the results of applying the given [transformer] function to the values\n * in the underlying [sequence], where the transformer function takes the index of the value in the underlying\n * sequence along with the value itself.\n */\ninternal class TransformingIndexedSequence<T, R>\nconstructor(private val sequence: Sequence<T>, private val transformer: (Int, T) -> R) : Sequence<R> {\n    override fun iterator(): Iterator<R> = object : Iterator<R> {\n        val iterator = sequence.iterator()\n        var index = 0\n        override fun next(): R {\n            return transformer(checkIndexOverflow(index++), iterator.next())\n        }\n    }\n\n    override fun hasNext(): Boolean {\n        return iterator.hasNext()\n    }\n}\n\n/**\n * A sequence which combines values from the underlying [sequence] with their indices and returns them as\n * [IndexedValue] objects.\n */\ninternal class IndexingSequence<T>\nconstructor(private val sequence: Sequence<T>) : Sequence<IndexedValue<T>> {\n    override fun iterator(): Iterator<IndexedValue<T>> = object : Iterator<IndexedValue<T>> {\n        val iterator = sequence.iterator()\n        var index = 0\n        override fun next(): IndexedValue<T> {\n            return IndexedValue(checkIndexOverflow(index++), iterator.next())\n        }\n    }\n\n    override fun hasNext(): Boolean {\n        return iterator.hasNext()\n    }\n}\n\n/**\n * A sequence which takes the values from two parallel underlying sequences, passes them to the given\n * [transform] function and returns the values returned by that function. The sequence stops returning\n * values as soon as one of the underlying sequences stops returning values.\n */\ninternal class MergingSequence<T1, T2, V>\nconstructor(\n    private val sequence1: Sequence<T1>,\n    private val sequence2: Sequence<T2>,\n    private val transform: (T1, T2) -> V) : Sequence<V> {\n    override fun iterator(): Iterator<V> = object : Iterator<V> {\n        val iterator1 = sequence1.iterator()\n        val iterator2 = sequence2.iterator()\n        override fun next(): V {\n            return transform(iterator1.next(), iterator2.next())\n        }\n    }\n\n    override fun hasNext(): Boolean {\n        return iterator1.hasNext() && iterator2.hasNext()\n    }\n}\n\ninternal class FlatteningSequence<T, R, E>\nconstructor(\n    private val sequence: Sequence<T>,\n    private val transformer: (T) -> R,\n    private val iterator: (R) -> Iterator<E>) : Sequence<E> {\n    override fun iterator(): Iterator<E> = object : Iterator<E> {\n        val iterator = sequence.iterator()\n        var itemIterator: Iterator<E>? = null\n        override fun next(): E {\n            if (!ensureItemIterator())\n                throw NoSuchElementException()\n            return itemIterator!!.next()\n        }\n    }\n\n    override fun hasNext(): Boolean {\n        return ensureItemIterator()\n    }\n\n    private fun ensureItemIterator(): Boolean {\n        if (itemIterator?.hasNext() == false)\n            itemIterator = null\n        while (itemIterator == null) {\n            if (!iterator.hasNext())\n                return false\n            } else {\n                val element = iterator.next()\n                val nextItemIterator = iterator(transformer(element))\n            }\n        }\n        if (nextItemIterator.hasNext())\n            itemIterator = nextItemIterator\n        return true\n    }\n}\n\ninternal fun <T, C, R> flatMapIndexed(source: Sequence<T>, transform: (Int, T) -> C, iterator: (C) -> Iterator<R>): Sequence<R> =\n    sequence {\n        var index = 0\n        for (element in source) {\n            val result = transform(checkIndexOverflow(index++), element)\n            yieldAll(iterator(result))\n        }\n    }\n\n/**\n * A sequence that supports drop(n) and take(n) operations\n */\ninternal interface DropTakeSequence<T> : Sequence<T> {\n    fun drop(n: Int): Sequence<T>\n    fun take(n:

```

```

Int): Sequence<T>\n\n/**\n * A sequence that skips [startIndex] values from the underlying [sequence]\n * and stops returning values right before [endIndex], i.e. stops at `endIndex - 1`\n */\ninternal class SubSequence<T>(\n private val sequence: Sequence<T>,\n private val startIndex: Int,\n private val endIndex: Int)\n) : Sequence<T>,\n DropTakeSequence<T> {\n\n init {\n require(startIndex >= 0) { \"startIndex should be non-negative, but is\n $startIndex\" }\n require(endIndex >= 0) { \"endIndex should be non-negative, but is $endIndex\" }\n require(endIndex >= startIndex) { \"endIndex should be not less than startIndex, but was $endIndex < $startIndex\" }\n }\n\n private val count: Int get() = endIndex - startIndex\n\n override fun drop(n: Int): Sequence<T> = if\n (>= count) emptySequence() else SubSequence(sequence, startIndex + n, endIndex)\n\n override fun take(n: Int): Sequence<T> = if (n >= count) this else SubSequence(sequence, startIndex, startIndex + n)\n\n\n override fun\n iterator(): object : Iterator<T> {\n\n val iterator = sequence.iterator()\n var position = 0\n\n //\n Shouldn't be called from constructor to avoid premature iteration\n private fun drop() {\n while (position\n < startIndex && iterator.hasNext()) {\n iterator.next()\n position++\n }\n }\n\n override fun hasNext(): Boolean {\n drop()\n return (position < endIndex) && iterator.hasNext()\n }\n\n override fun next(): T {\n drop()\n if (position >= endIndex)\n throw\n NoSuchElementException()\n position++\n return iterator.next()\n }\n }\n}\n\n/**\n * A\n sequence that returns at most [count] values from the underlying [sequence], and stops returning values\n * as soon as that count is reached.\n */\ninternal class TakeSequence<T>(\n private val sequence: Sequence<T>,\n private\n val count: Int)\n) : Sequence<T>,\n DropTakeSequence<T> {\n\n init {\n require(count >= 0) { \"count must be\n non-negative, but was $count.\" }\n }\n\n override fun drop(n: Int): Sequence<T> = if (n >= count)\n emptySequence() else SubSequence(sequence, n, count)\n\n override fun take(n: Int): Sequence<T> = if (n >=\n count) this else TakeSequence(sequence, n)\n\n\n override fun iterator(): Iterator<T> = object : Iterator<T> {\n\n var left = count\n val iterator = sequence.iterator()\n\n override fun next(): T {\n if (left == 0)\n throw NoSuchElementException()\n left--\n return iterator.next()\n }\n\n override fun\n hasNext(): Boolean {\n return left > 0 && iterator.hasNext()\n }\n }\n}\n\n/**\n * A sequence that\n returns values from the underlying [sequence] while the [predicate] function returns `true`, and stops returning\n values once the function returns `false` for the next element.\n */\ninternal class\n TakeWhileSequence<T>(\n constructor(\n private val sequence: Sequence<T>,\n private val predicate: (T) ->\n Boolean)\n) : Sequence<T> {\n\n override fun iterator(): Iterator<T> = object : Iterator<T> {\n\n val iterator =\n sequence.iterator()\n var nextState: Int = -1 // -1 for unknown, 0 for done, 1 for continue\n var nextItem: T?\n = null\n\n private fun calcNext() {\n if (iterator.hasNext()) {\n val item = iterator.next()\n\n if (predicate(item)) {\n nextState = 1\n nextItem = item\n return\n }\n\n nextState = 0\n }\n\n override fun next(): T {\n if (nextState == -1)\n calcNext() // will change nextState\n if (nextState == 0)\n throw NoSuchElementException()\n @Suppress(\"UNCHECKED_CAST\")\n val result = nextItem as T\n // Clean next to avoid keeping\n reference on yielded instance\n nextItem = null\n nextState = -1\n return result\n }\n\n override fun hasNext(): Boolean {\n if (nextState == -1)\n calcNext() // will change nextState\n return nextState == 1\n }\n }\n}\n\n/**\n * A sequence that skips the specified number of values from the\n underlying [sequence] and returns\n * all values after that.\n */\ninternal class DropSequence<T>(\n private val\n sequence: Sequence<T>,\n private val count: Int)\n) : Sequence<T>,\n DropTakeSequence<T> {\n\n init {\n require(count >= 0) { \"count must be non-negative, but was $count.\" }\n }\n\n override fun drop(n: Int): Sequence<T> = (count + n).let { n1 -> if (n1 < 0) DropSequence(this, n) else DropSequence(sequence, n1) }\n\n\n override fun take(n: Int): Sequence<T> = (count + n).let { n1 -> if (n1 < 0) TakeSequence(this, n) else\n SubSequence(sequence, count, n1) }\n\n\n override fun iterator(): Iterator<T> = object : Iterator<T> {\n\n val\n iterator = sequence.iterator()\n var left = count\n\n // Shouldn't be called from constructor to avoid\n premature iteration\n private fun drop() {\n while (left > 0 && iterator.hasNext()) {\n\n iterator.next()\n left--\n }\n }\n\n override fun next(): T {\n drop()\n return\n iterator.next()\n }\n\n override fun hasNext(): Boolean {\n drop()\n return iterator.hasNext()\n }\n }\n}\n\n/**\n * A sequence that skips the values from the underlying [sequence] while the given

```

```

[predicate] returns `true` and returns\n * all values after that.\n *\ninternal class
DropWhileSequence<T>\nconstructor(\n private val sequence: Sequence<T>,\n private val predicate: (T) ->
Boolean)\n) : Sequence<T> {\n\n override fun iterator(): Iterator<T> = object : Iterator<T> {\n val iterator =
sequence.iterator()\n var dropState: Int = -1 // -1 for not dropping, 1 for nextItem, 0 for normal iteration\n
var nextItem: T? = null\n\n private fun drop() {\n while (iterator.hasNext()) {\n val item =
iterator.next()\n if (!predicate(item)) {\n nextItem = item\n dropState = 1\n
return\n }\n }\n dropState = 0\n }\n\n override fun next(): T {\n if
(dropState == -1)\n drop()\n if (dropState == 1) {\n
@Suppress("UNCHECKED_CAST")\n val result = nextItem as T\n nextItem = null\n
dropState = 0\n return result\n }\n return iterator.next()\n }\n\n override fun
hasNext(): Boolean {\n if (dropState == -1)\n drop()\n return dropState == 1 ||
iterator.hasNext()\n }\n }\n}\n\ninternal class DistinctSequence<T, K>(private val source: Sequence<T>,\n
private val keySelector: (T) -> K) : Sequence<T> {\n override fun iterator(): Iterator<T> =
DistinctIterator(source.iterator(), keySelector)\n}\n\nprivate class DistinctIterator<T, K>(private val source:
Iterator<T>,\n private val keySelector: (T) -> K) : AbstractIterator<T>() {\n private val observed =
HashSet<K>()\n\n override fun computeNext() {\n while (source.hasNext()) {\n val next =
source.next()\n val key = keySelector(next)\n\n if (observed.add(key)) {\n setNext(next)\n
return\n }\n }\n done()\n }\n}\n\nprivate class GeneratorSequence<T : Any>(private val
getInitialValue: () -> T?,\n private val getNextValue: (T) -> T?) : Sequence<T> {\n override fun iterator():
Iterator<T> = object : Iterator<T> {\n var nextItem: T? = null\n var nextState: Int = -2 // -2 for initial
unknown, -1 for next unknown, 0 for done, 1 for continue\n\n private fun calcNext() {\n nextItem = if
(nextState == -2) getInitialValue() else getNextValue(nextItem!!)\n nextState = if (nextItem == null) 0 else
1\n }\n\n override fun next(): T {\n if (nextState < 0)\n calcNext()\n\n if (nextState
== 0)\n throw NoSuchElementException()\n val result = nextItem as T\n // Do not clean
nextItem (to avoid keeping reference on yielded instance) -- need to keep state for getNextValue\n nextState
= -1\n return result\n }\n\n override fun hasNext(): Boolean {\n if (nextState < 0)\n
calcNext()\n return nextState == 1\n }\n }\n}\n\n/**\n * Returns a wrapper sequence that provides
values of this sequence, but ensures it can be iterated only one time.\n *\n * The operation is _intermediate_ and
_stateless_.  

 * [IllegalStateException] is thrown on iterating the returned sequence for the second time and the
following times.\n *\n */\npublic fun <T> Sequence<T>.constrainOnce(): Sequence<T> {\n // as? does not work
in js\n //return this as? ConstrainedOnceSequence<T>?: ConstrainedOnceSequence(this)\n return if (this is
ConstrainedOnceSequence<T>) this else ConstrainedOnceSequence(this)\n}\n\n/**\n * Returns a sequence which
invokes the function to calculate the next value on each iteration until the function returns `null`.\n *\n * The
returned sequence is constrained to be iterated only once.\n *\n * @see constrainOnce\n *\n */\n@see
kotlin.sequences.sequence\n *\n */\n@sample samples.collections.Sequences.Building.generateSequence\n */\npublic
fun <T : Any> generateSequence(nextFunction: () -> T?): Sequence<T> {\n return
GeneratorSequence(nextFunction, { nextFunction() }).constrainOnce()\n}\n\n/**\n * Returns a sequence defined by
the starting value [seed] and the function [nextFunction],\n *\n * which is invoked to calculate the next value based on
the previous one on each iteration.\n *\n * The sequence produces values until it encounters first `null` value.\n *\n * If
[seed] is `null`, an empty sequence is produced.\n *\n * The sequence can be iterated multiple times, each time
starting with [seed].\n *\n * @see kotlin.sequences.sequence\n *\n */\n@sample
samples.collections.Sequences.Building.generateSequenceWithSeed\n\n */\n@kotlin.internal.LowPriorityInOverloadResolution\npublic fun <T : Any> generateSequence(seed: T?,\n
nextFunction: (T) -> T?): Sequence<T> =\n if (seed == null)\n EmptySequence\n else\n
GeneratorSequence({ seed }, nextFunction)\n\n/**\n * Returns a sequence defined by the function [seedFunction],
which is invoked to produce the starting value,\n *\n * and the [nextFunction], which is invoked to calculate the next
value based on the previous one on each iteration.\n *\n * The sequence produces values until it encounters first
`null` value.\n *\n * If [seedFunction] returns `null`, an empty sequence is produced.\n *\n * The sequence can be

```

```

iterated multiple times.\n *\n * @see kotlin.sequences.sequence\n *\n * @sample
samples.collections.Sequences.Building.generateSequenceWithLazySeed\n *\npublic fun <T : Any>
generateSequence(seedFunction: () -> T?, nextFunction: (T) -> T?): Sequence<T> =\n
GeneratorSequence(seedFunction, nextFunction)\n\n"/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin
Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.\n
*/\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("PreconditionsKt")\n\npackage
kotlin\n\nimport kotlin.contracts.contract\n\n/**\n * Throws an [IllegalArgumentException] if the [value] is false.\n
*\n * @sample samples.misc.Preconditions.failRequireWithLazyMessage\n *\n@kotlin.internal.InlineOnly\npublic
inline fun require(value: Boolean): Unit {\n    contract {\n        returns() implies value\n    }\n    require(value) {
\n        "Failed requirement.\n    }\n}\n\n/**\n * Throws an [IllegalArgumentException] with the result of calling
[lazyMessage] if the [value] is false.\n
*\n * @sample samples.misc.Preconditions.failRequireWithLazyMessage\n
*\n@kotlin.internal.InlineOnly\npublic inline fun require(value: Boolean, lazyMessage: () -> Any): Unit {\n
    contract {\n        returns() implies value\n    }\n    if (!value) {\n        val message = lazyMessage()\n        throw
    IllegalArgumentException(message.toString())\n    }\n}\n\n/**\n * Throws an [IllegalArgumentException] if the
[value] is null. Otherwise returns the not null value.\n
*\n@kotlin.internal.InlineOnly\npublic inline fun <T : Any>
requireNotNull(value: T?): T {\n    contract {\n        returns() implies (value != null)\n    }\n    return
requireNotNull(value) { "Required value was null.\n    }\n}\n\n/**\n * Throws an [IllegalArgumentException] with
the result of calling [lazyMessage] if the [value] is null. Otherwise\n * returns the not null value.\n
*\n * @sample
samples.misc.Preconditions.failRequireNotNullWithLazyMessage\n *\n@kotlin.internal.InlineOnly\npublic inline
fun <T : Any> requireNotNull(value: T?, lazyMessage: () -> Any): T {\n    contract {\n        returns() implies (value
!= null)\n    }\n    if (value == null) {\n        val message = lazyMessage()\n        throw
    IllegalArgumentException(message.toString())\n    } else {\n        return value\n    }\n}\n\n/**\n * Throws an
[IllegalStateException] if the [value] is false.\n
*\n * @sample
samples.misc.Preconditions.failCheckWithLazyMessage\n *\n@kotlin.internal.InlineOnly\npublic inline fun
check(value: Boolean): Unit {\n    contract {\n        returns() implies value\n    }\n    check(value) { "Check failed.\n
    }\n}\n\n/**\n * Throws an [IllegalStateException] with the result of calling [lazyMessage] if the [value] is false.\n
*\n * @sample samples.misc.Preconditions.failCheckWithLazyMessage\n *\n@kotlin.internal.InlineOnly\npublic
inline fun check(value: Boolean, lazyMessage: () -> Any): Unit {\n    contract {\n        returns() implies value\n    }\n
    if (!value) {\n        val message = lazyMessage()\n        throw IllegalStateException(message.toString())\n    }\n}\n\n/**\n * Throws an [IllegalStateException] if the [value] is null. Otherwise\n * returns the not null value.\n
*\n * @sample samples.misc.Preconditions.failCheckWithLazyMessage\n *\n@kotlin.internal.InlineOnly\npublic
inline fun <T : Any> checkNotNull(value: T?): T {\n    contract {\n        returns() implies (value != null)\n    }\n
    return checkNotNull(value) { "Required value was null.\n    }\n}\n\n/**\n * Throws an [IllegalStateException] with
the result of calling [lazyMessage] if the [value] is null. Otherwise\n * returns the not null value.\n
*\n * @sample
samples.misc.Preconditions.failCheckWithLazyMessage\n *\n@kotlin.internal.InlineOnly\npublic inline fun <T :
Any> checkNotNull(value: T?, lazyMessage: () -> Any): T {\n    contract {\n        returns() implies (value != null)\n
    }\n    if (value == null) {\n        val message = lazyMessage()\n        throw
    IllegalStateException(message.toString())\n    } else {\n        return value\n    }\n}\n\n/**\n * Throws an
[IllegalStateException] with the given [message].\n
*\n * @sample samples.misc.Preconditions.failWithError\n
*\n@kotlin.internal.InlineOnly\npublic inline fun error(message: Any): Nothing = throw
    IllegalStateException(message.toString())\n\n"/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n
*/\n\npackage kotlin.collections\n\n/\n\n// NOTE: THIS FILE IS AUTO-GENERATED
by the GenerateStandardLib.kt\n// See: https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\nimport
kotlin.js.*\nimport primitiveArrayConcat\nimport withType\nimport kotlin.ranges.contains\nimport
kotlin.ranges.reversed\n\n/**\n * Returns an element at the given [index] or throws an
[IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n
*\n *\n * @sample

```

```

samples.collections.Collections.Elements.elementAt\n *^npublic actual fun <T> Array<out T>.elementAt(index:
Int): T {\n  return elementAtOrElse(index) { throw IndexOutOfBoundsException("\index: $index, size: $size}")
}\n}\n\n/**\n * Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is
out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.elementAt\n *^npublic
actual fun ByteArray.elementAt(index: Int): Byte {\n  return elementAtOrElse(index) { throw
IndexOutOfBoundsException("\index: $index, size: $size}") }\n}\n\n/**\n * Returns an element at the given
[index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAt\n *^npublic actual fun ShortArray.elementAt(index: Int): Short
{\n  return elementAtOrElse(index) { throw IndexOutOfBoundsException("\index: $index, size: $size}")
}\n}\n\n/**\n * Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is
out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.elementAt\n *^npublic
actual fun IntArray.elementAt(index: Int): Int {\n  return elementAtOrElse(index) { throw
IndexOutOfBoundsException("\index: $index, size: $size}") }\n}\n\n/**\n * Returns an element at the given
[index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAt\n *^npublic actual fun LongArray.elementAt(index: Int): Long
{\n  return elementAtOrElse(index) { throw IndexOutOfBoundsException("\index: $index, size: $size}")
}\n}\n\n/**\n * Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is
out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.elementAt\n *^npublic
actual fun FloatArray.elementAt(index: Int): Float {\n  return elementAtOrElse(index) { throw
IndexOutOfBoundsException("\index: $index, size: $size}") }\n}\n\n/**\n * Returns an element at the given
[index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAt\n *^npublic actual fun DoubleArray.elementAt(index: Int):
Double {\n  return elementAtOrElse(index) { throw IndexOutOfBoundsException("\index: $index, size: $size}")
}\n}\n\n/**\n * Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is
out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.elementAt\n *^npublic
actual fun BooleanArray.elementAt(index: Int): Boolean {\n  return elementAtOrElse(index) { throw
IndexOutOfBoundsException("\index: $index, size: $size}") }\n}\n\n/**\n * Returns an element at the given
[index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAt\n *^npublic actual fun CharArray.elementAt(index: Int): Char
{\n  return elementAtOrElse(index) { throw IndexOutOfBoundsException("\index: $index, size: $size}")
}\n}\n\n/**\n * Returns a [List] that wraps the original array.\n *^npublic actual fun <T> Array<out T>.asList():
List<T> {\n  return ArrayList<T>(this.unsafeCast<Array<Any?>>())\n}\n\n/**\n * Returns a [List] that wraps the
original array.\n *^n@kotlin.internal.InlineOnly\npublic actual inline fun ByteArray.asList(): List<Byte> {\n
return this.unsafeCast<Array<Byte>>().asList()\n}\n\n/**\n * Returns a [List] that wraps the original array.\n
*^n@kotlin.internal.InlineOnly\npublic actual inline fun ShortArray.asList(): List<Short> {\n  return
this.unsafeCast<Array<Short>>().asList()\n}\n\n/**\n * Returns a [List] that wraps the original array.\n
*^n@kotlin.internal.InlineOnly\npublic actual inline fun IntArray.asList(): List<Int> {\n  return
this.unsafeCast<Array<Int>>().asList()\n}\n\n/**\n * Returns a [List] that wraps the original array.\n
*^n@kotlin.internal.InlineOnly\npublic actual inline fun LongArray.asList(): List<Long> {\n  return
this.unsafeCast<Array<Long>>().asList()\n}\n\n/**\n * Returns a [List] that wraps the original array.\n
*^n@kotlin.internal.InlineOnly\npublic actual inline fun FloatArray.asList(): List<Float> {\n  return
this.unsafeCast<Array<Float>>().asList()\n}\n\n/**\n * Returns a [List] that wraps the original array.\n
*^n@kotlin.internal.InlineOnly\npublic actual inline fun DoubleArray.asList(): List<Double> {\n  return
this.unsafeCast<Array<Double>>().asList()\n}\n\n/**\n * Returns a [List] that wraps the original array.\n
*^n@kotlin.internal.InlineOnly\npublic actual inline fun BooleanArray.asList(): List<Boolean> {\n  return
this.unsafeCast<Array<Boolean>>().asList()\n}\n\n/**\n * Returns a [List] that wraps the original array.\n
*^npublic actual fun CharArray.asList(): List<Char> {\n  return object : AbstractList<Char>(), RandomAccess {\n
override val size: Int get() = this@asList.size\n      override fun isEmpty(): Boolean = this@asList.isEmpty()\n

```

```

    override fun contains(element: Char): Boolean = this@asList.contains(element)\n
Char {\n
    AbstractList.checkElementIndex(index, size)\n
    return this@asList[index]\n
}\n
override fun indexOf(element: Char): Int {\n
    @Suppress("USELESS_CAST")\n
    if ((element as Any?) !is Char) return -1\n
    return this@asList.indexOf(element)\n
}\n
override fun
lastIndexOf(element: Char): Int {\n
    @Suppress("USELESS_CAST")\n
    if ((element as Any?) !is Char) return -1\n
    return this@asList.lastIndexOf(element)\n
}\n
}\n}\n\n/**\n * Returns `true` if the
two specified arrays are *deeply* equal to one another,\n * i.e. contain the same number of the same elements in the
same order.\n * \n * If two corresponding elements are nested arrays, they are also compared deeply.\n * If any of
arrays contains itself on any nesting level the behavior is undefined.\n * \n * The elements of other types are
compared for equality with the [equals][Any.equals] function.\n * For floating point numbers it means that `NaN` is
equal to itself and `-0.0` is not equal to `0.0`.\n
*/\n\n@SinceKotlin("1.1")\n@kotlin.internal.LowPriorityInOverloadResolution\npublic actual infix fun <T>
Array<out T>.contentDeepEquals(other: Array<out T>): Boolean {\n
    return
this.contentDeepEquals(other)\n}\n\n/**\n * Returns `true` if the two specified arrays are *deeply* equal to one
another,\n * i.e. contain the same number of the same elements in the same order.\n * \n * The specified arrays are
also considered deeply equal if both are `null`.\n * \n * If two corresponding elements are nested arrays, they are
also compared deeply.\n * If any of arrays contains itself on any nesting level the behavior is undefined.\n * \n * The
elements of other types are compared for equality with the [equals][Any.equals] function.\n * For floating point
numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n
*/\n\n@SinceKotlin("1.4")\n@library("arrayDeepEquals")\npublic actual infix fun <T> Array<out
T>?.contentDeepEquals(other: Array<out T>?): Boolean {\n
    definedExternally\n}\n\n/**\n * Returns a hash code
based on the contents of this array as if it is [List].\n * Nested arrays are treated as lists too.\n * \n * If any of
arrays contains itself on any nesting level the behavior is undefined.\n
*/\n\n@SinceKotlin("1.1")\n@kotlin.internal.LowPriorityInOverloadResolution\npublic actual fun <T> Array<out
T>.contentDeepHashCode(): Int {\n
    return this.contentDeepHashCode()\n}\n\n/**\n * Returns a hash code based
on the contents of this array as if it is [List].\n * Nested arrays are treated as lists too.\n * \n * If any of
arrays contains itself on any nesting level the behavior is undefined.\n
*/\n\n@SinceKotlin("1.4")\n@library("arrayDeepHashCode")\npublic actual fun <T> Array<out
T>?.contentDeepHashCode(): Int {\n
    definedExternally\n}\n\n/**\n * Returns a string
representation of the
contents of this array as if it is a [List].\n * Nested arrays are treated as lists too.\n * \n * If any of
arrays contains itself on any nesting level that reference\n * is rendered as `"[...]"` to prevent recursion.\n * \n *
@sample samples.collections.Arrays.ContentOperations.contentDeepToString\n
*/\n\n@SinceKotlin("1.1")\n@kotlin.internal.LowPriorityInOverloadResolution\npublic actual fun <T> Array<out
T>.contentDeepToString(): String {\n
    return this.contentDeepToString()\n}\n\n/**\n * Returns a string
representation of the contents of this array as if it is a [List].\n * Nested arrays are treated as lists too.\n * \n *
If any of arrays contains itself on any nesting level that reference\n * is rendered as `"[...]"` to prevent recursion.\n
*/\n\n@sample samples.collections.Arrays.ContentOperations.contentDeepToString\n
*/\n\n@SinceKotlin("1.4")\n@library("arrayDeepToString")\npublic actual fun <T> Array<out
T>?.contentDeepToString(): String {\n
    definedExternally\n}\n\n/**\n * Returns `true` if the two specified arrays
are *structurally* equal to one another,\n * i.e. contain the same number of the same elements in the same order.\n
*/\n\n * The elements are compared for equality with the [equals][Any.equals] function.\n * For floating point
numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic
actual infix fun <T> Array<out T>.contentEquals(other: Array<out T>): Boolean {\n
    return
this.contentEquals(other)\n}\n\n/**\n * Returns `true` if the two specified arrays are *structurally* equal to one
another,\n * i.e. contain the same number of the same elements in the same order.\n * \n * The elements are
compared for equality with the [equals][Any.equals] function.\n * For floating point numbers it means that `NaN` is
equal to itself and `-0.0` is not equal to `0.0`.\n\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation

```

```

warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic actual infix fun
ByteArray.contentEquals(other: ByteArray): Boolean {\n    return this.contentEquals(other)\n}\n\n/**\n * Returns
`true` if the two specified arrays are *structurally* equal to one another,\n * i.e. contain the same number of the
same elements in the same order.\n * \n * The elements are compared for equality with the [equals][Any.equals]
function.\n * For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n
*\n*\n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic actual infix fun
ShortArray.contentEquals(other: ShortArray): Boolean {\n    return this.contentEquals(other)\n}\n\n/**\n * Returns
`true` if the two specified arrays are *structurally* equal to one another,\n * i.e. contain the same number of the
same elements in the same order.\n * \n * The elements are compared for equality with the [equals][Any.equals]
function.\n * For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n
*\n*\n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic actual infix fun
IntArray.contentEquals(other: IntArray): Boolean {\n    return this.contentEquals(other)\n}\n\n/**\n * Returns
`true` if the two specified arrays are *structurally* equal to one another,\n * i.e. contain the same number of the same
elements in the same order.\n * \n * The elements are compared for equality with the [equals][Any.equals]
function.\n * For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n
*\n*\n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic actual infix fun
LongArray.contentEquals(other: LongArray): Boolean {\n    return this.contentEquals(other)\n}\n\n/**\n * Returns
`true` if the two specified arrays are *structurally* equal to one another,\n * i.e. contain the same number of the
same elements in the same order.\n * \n * The elements are compared for equality with the [equals][Any.equals]
function.\n * For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n
*\n*\n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic actual infix fun
FloatArray.contentEquals(other: FloatArray): Boolean {\n    return this.contentEquals(other)\n}\n\n/**\n * Returns
`true` if the two specified arrays are *structurally* equal to one another,\n * i.e. contain the same number of the
same elements in the same order.\n * \n * The elements are compared for equality with the [equals][Any.equals]
function.\n * For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n
*\n*\n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic actual infix fun
DoubleArray.contentEquals(other: DoubleArray): Boolean {\n    return this.contentEquals(other)\n}\n\n/**\n *
Returns `true` if the two specified arrays are *structurally* equal to one another,\n * i.e. contain the same number of
the same elements in the same order.\n * \n * The elements are compared for equality with the [equals][Any.equals]
function.\n * For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n
*\n*\n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic actual infix fun
BooleanArray.contentEquals(other: BooleanArray): Boolean {\n    return this.contentEquals(other)\n}\n\n/**\n *
Returns `true` if the two specified arrays are *structurally* equal to one another,\n * i.e. contain the same number of
the same elements in the same order.\n * \n * The elements are compared for equality with the [equals][Any.equals]
function.\n * For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n
*\n*\n@Deprecated(\"Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin(\"1.1\")\n@DeprecatedSinceKotlin(hiddenSince = \"1.4\")\npublic actual infix fun
CharArray.contentEquals(other: CharArray): Boolean {\n    return this.contentEquals(other)\n}\n\n/**\n * Returns
`true` if the two specified arrays are *structurally* equal to one another,\n * i.e. contain the same number of the
same elements in the same order.\n * \n * The elements are compared for equality with the [equals][Any.equals]
function.\n * For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.\n
*\n*\n@SinceKotlin(\"1.4\")\n@library(\"arrayEquals\")\npublic actual infix fun <T> Array<out

```

T>?.contentEquals(other: Array<out T>?): Boolean {  
definedExternally  
Returns `true` if the two specified arrays are \*structurally\* equal to one another, i.e. contain the same number of the same elements in the same order.  
The elements are compared for equality with the [equals][Any.equals] function.  
For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.

```
*\n@SinceKotlin("1.4")\n@library("arrayEquals")\npublic actual infix fun ByteArray?.contentEquals(other: ByteArray?): Boolean {  
definedExternally  
Returns `true` if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order.  
The elements are compared for equality with the [equals][Any.equals] function.  
For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.
```

```
*\n@SinceKotlin("1.4")\n@library("arrayEquals")\npublic actual infix fun ShortArray?.contentEquals(other: ShortArray?): Boolean {  
definedExternally  
Returns `true` if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order.  
The elements are compared for equality with the [equals][Any.equals] function.  
For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.
```

```
*\n@SinceKotlin("1.4")\n@library("arrayEquals")\npublic actual infix fun IntArray?.contentEquals(other: IntArray?): Boolean {  
definedExternally  
Returns `true` if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order.  
The elements are compared for equality with the [equals][Any.equals] function.  
For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.
```

```
*\n@SinceKotlin("1.4")\n@library("arrayEquals")\npublic actual infix fun LongArray?.contentEquals(other: LongArray?): Boolean {  
definedExternally  
Returns `true` if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order.  
The elements are compared for equality with the [equals][Any.equals] function.  
For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.
```

```
*\n@SinceKotlin("1.4")\n@library("arrayEquals")\npublic actual infix fun FloatArray?.contentEquals(other: FloatArray?): Boolean {  
definedExternally  
Returns `true` if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order.  
The elements are compared for equality with the [equals][Any.equals] function.  
For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.
```

```
*\n@SinceKotlin("1.4")\n@library("arrayEquals")\npublic actual infix fun DoubleArray?.contentEquals(other: DoubleArray?): Boolean {  
definedExternally  
Returns `true` if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order.  
The elements are compared for equality with the [equals][Any.equals] function.  
For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.
```

```
*\n@SinceKotlin("1.4")\n@library("arrayEquals")\npublic actual infix fun BooleanArray?.contentEquals(other: BooleanArray?): Boolean {  
definedExternally  
Returns `true` if the two specified arrays are *structurally* equal to one another, i.e. contain the same number of the same elements in the same order.  
The elements are compared for equality with the [equals][Any.equals] function.  
For floating point numbers it means that `NaN` is equal to itself and `-0.0` is not equal to `0.0`.
```

```
*\n@SinceKotlin("1.4")\n@library("arrayEquals")\npublic actual infix fun CharArray?.contentEquals(other: CharArray?): Boolean {  
definedExternally  
Returns a hash code based on the contents of this array as if it is [List].  
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")
```

```
@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic actual fun <T> Array<out T>.contentHashCode(): Int {  
return this.contentHashCode()  
Returns a hash code based on the contents of this array as if it is [List].  
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")
```

```
@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic actual fun ByteArray.contentHashCode(): Int {  
return this.contentHashCode()  
Returns a hash code based on the contents of this array as if it is [List].  
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")
```

```

warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic actual fun
ShortArray.contentHashCode(): Int {\n    return this.contentHashCode()\n}\n\n/**\n * Returns a hash code based on
the contents of this array as if it is [List].\n */\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic actual fun
IntArray.contentHashCode(): Int {\n    return this.contentHashCode()\n}\n\n/**\n * Returns a hash code based on
the contents of this array as if it is [List].\n */\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic actual fun
LongArray.contentHashCode(): Int {\n    return this.contentHashCode()\n}\n\n/**\n * Returns a hash code based on
the contents of this array as if it is [List].\n */\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic actual fun
FloatArray.contentHashCode(): Int {\n    return this.contentHashCode()\n}\n\n/**\n * Returns a hash code based on
the contents of this array as if it is [List].\n */\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic actual fun
DoubleArray.contentHashCode(): Int {\n    return this.contentHashCode()\n}\n\n/**\n * Returns a hash code based
on the contents of this array as if it is [List].\n */\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic actual fun
BooleanArray.contentHashCode(): Int {\n    return this.contentHashCode()\n}\n\n/**\n * Returns a hash code based
on the contents of this array as if it is [List].\n */\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic actual fun
CharArray.contentHashCode(): Int {\n    return this.contentHashCode()\n}\n\n/**\n * Returns a hash code based on
the contents of this array as if it is [List].\n */\n@SinceKotlin("1.4")\n@library("arrayHashCode")\npublic actual
fun <T> Array<out T>?.contentHashCode(): Int {\n    definedExternally\n}\n\n/**\n * Returns a hash code based on
the contents of this array as if it is [List].\n */\n@SinceKotlin("1.4")\n@library("arrayHashCode")\npublic actual
fun ByteArray?.contentHashCode(): Int {\n    definedExternally\n}\n\n/**\n * Returns a hash code based on the
contents of this array as if it is [List].\n */\n@SinceKotlin("1.4")\n@library("arrayHashCode")\npublic actual fun
ShortArray?.contentHashCode(): Int {\n    definedExternally\n}\n\n/**\n * Returns a hash code based on the
contents of this array as if it is [List].\n */\n@SinceKotlin("1.4")\n@library("arrayHashCode")\npublic actual fun
IntArray?.contentHashCode(): Int {\n    definedExternally\n}\n\n/**\n * Returns a hash code based on the contents
of this array as if it is [List].\n */\n@SinceKotlin("1.4")\n@library("arrayHashCode")\npublic actual fun
LongArray?.contentHashCode(): Int {\n    definedExternally\n}\n\n/**\n * Returns a hash code based on the
contents of this array as if it is [List].\n */\n@SinceKotlin("1.4")\n@library("arrayHashCode")\npublic actual fun
FloatArray?.contentHashCode(): Int {\n    definedExternally\n}\n\n/**\n * Returns a hash code based on the
contents of this array as if it is [List].\n */\n@SinceKotlin("1.4")\n@library("arrayHashCode")\npublic actual fun
DoubleArray?.contentHashCode(): Int {\n    definedExternally\n}\n\n/**\n * Returns a hash code based on the
contents of this array as if it is [List].\n */\n@SinceKotlin("1.4")\n@library("arrayHashCode")\npublic actual fun
BooleanArray?.contentHashCode(): Int {\n    definedExternally\n}\n\n/**\n * Returns a hash code based on the
contents of this array as if it is [List].\n */\n@SinceKotlin("1.4")\n@library("arrayHashCode")\npublic actual fun
CharArray?.contentHashCode(): Int {\n    definedExternally\n}\n\n/**\n * Returns a string representation of the
contents of the specified array as if it is [List].\n */\n\n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n */\n@Deprecated("Use Kotlin compiler 1.4 to
avoid deprecation warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic
actual fun <T> Array<out T>.contentToString(): String {\n    return this.contentToString()\n}\n\n/**\n * Returns a
string representation of the contents of the specified array as if it is [List].\n */\n\n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n */\n@Deprecated("Use Kotlin compiler 1.4 to
avoid deprecation warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic
actual fun ByteArray.contentToString(): String {\n    return this.contentToString()\n}\n\n/**\n * Returns a string
representation of the contents of the specified array as if it is [List].\n */\n\n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n */\n@Deprecated("Use Kotlin compiler 1.4 to

```

```

avoid deprecation warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic
actual fun ShortArray.contentToString(): String {\n    return this.contentToString()\n}\n\n/**\n * Returns a string
representation of the contents of the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n */\n@Deprecated("Use Kotlin compiler 1.4
to avoid deprecation warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic
actual fun IntArray.contentToString(): String {\n    return this.contentToString()\n}\n\n/**\n * Returns a string
representation of the contents of the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n */\n@Deprecated("Use Kotlin compiler 1.4
to avoid deprecation warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic
actual fun LongArray.contentToString(): String {\n    return this.contentToString()\n}\n\n/**\n * Returns a string
representation of the contents of the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n */\n@Deprecated("Use Kotlin compiler 1.4
to avoid deprecation warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic
actual fun FloatArray.contentToString(): String {\n    return this.contentToString()\n}\n\n/**\n * Returns a string
representation of the contents of the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n */\n@Deprecated("Use Kotlin compiler 1.4
to avoid deprecation warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic
actual fun DoubleArray.contentToString(): String {\n    return this.contentToString()\n}\n\n/**\n * Returns a string
representation of the contents of the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n */\n@Deprecated("Use Kotlin compiler 1.4
to avoid deprecation warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic
actual fun BooleanArray.contentToString(): String {\n    return this.contentToString()\n}\n\n/**\n * Returns a string
representation of the contents of the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n */\n@Deprecated("Use Kotlin compiler 1.4
to avoid deprecation warning.\")\n@SinceKotlin("1.1")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\npublic
actual fun CharArray.contentToString(): String {\n    return this.contentToString()\n}\n\n/**\n * Returns a string
representation of the contents of the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n\n */\n@SinceKotlin("1.4")\n@library("arrayToString")\npublic actual fun <T> Array<out T>?.contentToString():
String {\n    definedExternally\n}\n\n/**\n * Returns a string representation of the contents of the specified array as
if it is [List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n\n */\n@SinceKotlin("1.4")\n@library("arrayToString")\npublic actual fun ByteArray?.contentToString(): String
{\n    definedExternally\n}\n\n/**\n * Returns a string representation of the contents of the specified array as if it is
[List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n\n */\n@SinceKotlin("1.4")\n@library("arrayToString")\npublic actual fun ShortArray?.contentToString(): String
{\n    definedExternally\n}\n\n/**\n * Returns a string representation of the contents of the specified array as if it is
[List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n\n */\n@SinceKotlin("1.4")\n@library("arrayToString")\npublic actual fun IntArray?.contentToString(): String {\n    definedExternally\n}\n\n/**\n * Returns a string representation of the contents of the specified array as if it is
[List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n\n */\n@SinceKotlin("1.4")\n@library("arrayToString")\npublic actual fun LongArray?.contentToString(): String
{\n    definedExternally\n}\n\n/**\n * Returns a string representation of the contents of the specified array as if it is
[List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n\n */\n@SinceKotlin("1.4")\n@library("arrayToString")\npublic actual fun FloatArray?.contentToString(): String
{\n    definedExternally\n}\n\n/**\n * Returns a string representation of the contents of the specified array as if it is
[List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n\n */\n@SinceKotlin("1.4")\n@library("arrayToString")\npublic actual fun DoubleArray?.contentToString(): String
{\n    definedExternally\n}\n\n/**\n * Returns a string representation of the contents of the specified array as if it is

```

```

[List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n
*\n@SinceKotlin("1.4")\n@library("arrayToString")\npublic actual fun BooleanArray?.contentToString():
String {\n  definedExternally\n}\n\n/**\n * Returns a string representation of the contents of the specified array as
if it is [List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n
*\n@SinceKotlin("1.4")\n@library("arrayToString")\npublic actual fun CharArray?.contentToString(): String
{\n  definedExternally\n}\n\n/**\n * Copies this array or its subrange into the [destination] array and returns that
array.\n * \n * It's allowed to pass the same array in the [destination] and even specify the subrange so that it
overlaps with the destination range.\n * \n * @param destination the array to copy to.\n * @param destinationOffset
the position in the [destination] array to copy to, 0 by default.\n * @param startIndex the beginning (inclusive) of
the subrange to copy, 0 by default.\n * @param endIndex the end (exclusive) of the subrange to copy, size of this
array by default.\n * \n * @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex]
or [endIndex] is out of range of this array indices or when `startIndex > endIndex`.\n * @throws
IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified
[destinationOffset],\n * or when that index is out of the [destination] array indices range.\n * \n * @return the
[destination] array.\n
*\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT
_ARGUMENTS")\npublic actual inline fun <T> Array<out T>.copyInto(destination: Array<T>, destinationOffset:
Int = 0, startIndex: Int = 0, endIndex: Int = size): Array<T> {\n  arrayCopy(this, destination, destinationOffset,
startIndex, endIndex)\n  return destination\n}\n\n/**\n * Copies this array or its subrange into the [destination]
array and returns that array.\n * \n * It's allowed to pass the same array in the [destination] and even specify the
subrange so that it overlaps with the destination range.\n * \n * @param destination the array to copy to.\n *
@param destinationOffset the position in the [destination] array to copy to, 0 by default.\n * @param startIndex the
beginning (inclusive) of the subrange to copy, 0 by default.\n * @param endIndex the end (exclusive) of the
subrange to copy, size of this array by default.\n * \n * @throws IndexOutOfBoundsException or
[IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex
> endIndex`.\n * @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array
starting at the specified [destinationOffset],\n * or when that index is out of the [destination] array indices
range.\n * \n * @return the [destination] array.\n
*\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT
_ARGUMENTS")\npublic actual inline fun ByteArray.copyInto(destination: ByteArray, destinationOffset: Int = 0,
startIndex: Int = 0, endIndex: Int = size): ByteArray {\n  arrayCopy(this.unsafeCast<Array<Byte>>(),
destination.unsafeCast<Array<Byte>>(), destinationOffset, startIndex, endIndex)\n  return destination\n}\n\n/**\n
* Copies this array or its subrange into the [destination] array and returns that array.\n * \n * It's allowed to pass
the same array in the [destination] and even specify the subrange so that it overlaps with the destination range.\n
* \n * @param destination the array to copy to.\n * @param destinationOffset the position in the [destination] array
to copy to, 0 by default.\n * @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default.\n
* @param endIndex the end (exclusive) of the subrange to copy, size of this array by default.\n * \n * @throws
IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this
array indices or when `startIndex > endIndex`.\n * @throws IndexOutOfBoundsException when the subrange
doesn't fit into the [destination] array starting at the specified [destinationOffset],\n * or when that index is
out of the [destination] array indices range.\n * \n * @return the [destination] array.\n
*\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT
_ARGUMENTS")\npublic actual inline fun ShortArray.copyInto(destination: ShortArray, destinationOffset: Int =
0, startIndex: Int = 0, endIndex: Int = size): ShortArray {\n  arrayCopy(this.unsafeCast<Array<Short>>(),
destination.unsafeCast<Array<Short>>(), destinationOffset, startIndex, endIndex)\n  return destination\n}\n\n/**\n
* Copies this array or its subrange into the [destination] array and returns that array.\n * \n * It's allowed to pass
the same array in the [destination] and even specify the subrange so that it overlaps with the destination range.\n
* \n * @param destination the array to copy to.\n * @param destinationOffset the position in the [destination] array
to

```

copy to, 0 by default.\n \* @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default.\n \* @param endIndex the end (exclusive) of the subrange to copy, size of this array by default.\n \* \n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`.\n \* @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],\n \* or when that index is out of the [destination] array indices range.\n \* \n \* @return the [destination] array.\n

```
*\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic actual inline fun IntArray.copyInto(destination: IntArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size): IntArray {\n    arrayCopy(this.unsafeCast<Array<Int>>(), destination.unsafeCast<Array<Int>>(), destinationOffset, startIndex, endIndex)\n    return destination\n}\n\n/**\n * Copies this array or its subrange into the [destination] array and returns that array.\n * \n * It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range.\n * \n * @param destination the array to copy to.\n * @param destinationOffset the position in the [destination] array to copy to, 0 by default.\n * @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default.\n * @param endIndex the end (exclusive) of the subrange to copy, size of this array by default.\n * \n * @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`.\n * @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],\n * or when that index is out of the [destination] array indices range.\n * \n * @return the [destination] array.\n
```

```
*\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic actual inline fun LongArray.copyInto(destination: LongArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size): LongArray {\n    arrayCopy(this.unsafeCast<Array<Long>>(), destination.unsafeCast<Array<Long>>(), destinationOffset, startIndex, endIndex)\n    return destination\n}\n\n/**\n * Copies this array or its subrange into the [destination] array and returns that array.\n * \n * It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range.\n * \n * @param destination the array to copy to.\n * @param destinationOffset the position in the [destination] array to copy to, 0 by default.\n * @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default.\n * @param endIndex the end (exclusive) of the subrange to copy, size of this array by default.\n * \n * @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`.\n * @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],\n * or when that index is out of the [destination] array indices range.\n * \n * @return the [destination] array.\n
```

```
*\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic actual inline fun FloatArray.copyInto(destination: FloatArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size): FloatArray {\n    arrayCopy(this.unsafeCast<Array<Float>>(), destination.unsafeCast<Array<Float>>(), destinationOffset, startIndex, endIndex)\n    return destination\n}\n\n/**\n * Copies this array or its subrange into the [destination] array and returns that array.\n * \n * It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range.\n * \n * @param destination the array to copy to.\n * @param destinationOffset the position in the [destination] array to copy to, 0 by default.\n * @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default.\n * @param endIndex the end (exclusive) of the subrange to copy, size of this array by default.\n * \n * @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`.\n * @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],\n * or when that index is out of the [destination] array indices range.\n * \n * @return the [destination] array.\n
```

```
*\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic actual inline fun DoubleArray.copyInto(destination: DoubleArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size): DoubleArray {\n    arrayCopy(this.unsafeCast<Array<Double>>(),
```

```

destination.unsafeCast<Array<Double>>(), destinationOffset, startIndex, endIndex)\n    return
destination\n}\n\n/**\n * Copies this array or its subrange into the [destination] array and returns that array.\n * \n *
It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the
destination range.\n * \n * @param destination the array to copy to.\n * @param destinationOffset the position in the
[destination] array to copy to, 0 by default.\n * @param startIndex the beginning (inclusive) of the subrange to copy,
0 by default.\n * @param endIndex the end (exclusive) of the subrange to copy, size of this array by default.\n * \n *
@throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of
range of this array indices or when `startIndex > endIndex`.\n * @throws IndexOutOfBoundsException when the
subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],\n * or when that index is
out of the [destination] array indices range.\n * \n * @return the [destination] array.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT
_ARGUMENTS")\npublic actual inline fun BooleanArray.copyInto(destination: BooleanArray, destinationOffset:
Int = 0, startIndex: Int = 0, endIndex: Int = size): BooleanArray {\n
arrayCopy(this.unsafeCast<Array<Boolean>>(), destination.unsafeCast<Array<Boolean>>(), destinationOffset,
startIndex, endIndex)\n    return destination\n}\n\n/**\n * Copies this array or its subrange into the [destination]
array and returns that array.\n * \n * It's allowed to pass the same array in the [destination] and even specify the
subrange so that it overlaps with the destination range.\n * \n * @param destination the array to copy to.\n *
@param destinationOffset the position in the [destination] array to copy to, 0 by default.\n * @param startIndex the
beginning (inclusive) of the subrange to copy, 0 by default.\n * @param endIndex the end (exclusive) of the
subrange to copy, size of this array by default.\n * \n * @throws IndexOutOfBoundsException or
[IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex
> endIndex`.\n * @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array
starting at the specified [destinationOffset],\n * or when that index is out of the [destination] array indices range.\n *
\n * @return the [destination] array.\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT
_ARGUMENTS")\npublic actual inline fun CharArray.copyInto(destination: CharArray, destinationOffset: Int = 0,
startIndex: Int = 0, endIndex: Int = size): CharArray {\n    arrayCopy(this.unsafeCast<Array<Char>>(),
destination.unsafeCast<Array<Char>>(), destinationOffset, startIndex, endIndex)\n    return destination\n}\n\n/**\n *
Returns new array which is a copy of the original array.\n * \n * @sample
samples.collections.Arrays.CopyOfOperations.copyOf\n */\n@Suppress("ACTUAL_WITHOUT_EXPECT",
"NOTHING_TO_INLINE")\npublic actual inline fun <T> Array<out T>.copyOf(): Array<T> {\n    return
this.asDynamic().slice()\n}\n\n/**\n * Returns new array which is a copy of the original array.\n * \n * @sample
samples.collections.Arrays.CopyOfOperations.copyOf\n */\n@Suppress("NOTHING_TO_INLINE")\npublic
actual inline fun ByteArray.copyOf(): ByteArray {\n    return this.asDynamic().slice()\n}\n\n/**\n * Returns new
array which is a copy of the original array.\n * \n * @sample
samples.collections.Arrays.CopyOfOperations.copyOf\n */\n@Suppress("NOTHING_TO_INLINE")\npublic
actual inline fun ShortArray.copyOf(): ShortArray {\n    return this.asDynamic().slice()\n}\n\n/**\n * Returns new
array which is a copy of the original array.\n * \n * @sample
samples.collections.Arrays.CopyOfOperations.copyOf\n */\n@Suppress("NOTHING_TO_INLINE")\npublic
actual inline fun IntArray.copyOf(): IntArray {\n    return this.asDynamic().slice()\n}\n\n/**\n * Returns new array
which is a copy of the original array.\n * \n * @sample samples.collections.Arrays.CopyOfOperations.copyOf\n
*/\n@Suppress("NOTHING_TO_INLINE")\npublic
actual inline fun LongArray.copyOf(): LongArray {\n    return withType("LongArray",
this.asDynamic().slice()\n}\n\n/**\n * Returns new array which is a copy of the original array.\n * \n * @sample
samples.collections.Arrays.CopyOfOperations.copyOf\n */\n@Suppress("NOTHING_TO_INLINE")\npublic
actual inline fun FloatArray.copyOf(): FloatArray {\n    return this.asDynamic().slice()\n}\n\n/**\n * Returns new
array which is a copy of the original array.\n * \n * @sample
samples.collections.Arrays.CopyOfOperations.copyOf\n */\n@Suppress("NOTHING_TO_INLINE")\npublic
actual inline fun DoubleArray.copyOf(): DoubleArray {\n    return this.asDynamic().slice()\n}\n\n/**\n * Returns

```

```

new array which is a copy of the original array.\n * \n * @sample
samples.collections.Arrays.CopyOfOperations.copyOfOf\n * \n\npublic actual fun BooleanArray.copyOfOf():
BooleanArray {\n    return withType("<code>BooleanArray</code>", this.asDynamic().slice())\n}\n\n/**\n * Returns new array
which is a copy of the original array.\n * \n * @sample samples.collections.Arrays.CopyOfOperations.copyOfOf\n
* \n\npublic actual fun CharArray.copyOfOf(): CharArray {\n    return withType("<code>CharArray</code>",
this.asDynamic().slice())\n}\n\n/**\n * Returns new array which is a copy of the original array, resized to the given
[newSize].\n * The copy is either truncated or padded at the end with zero values if necessary.\n * \n * - If [newSize]
is less than the size of the original array, the copy array is truncated to the [newSize].\n * - If [newSize] is greater
than the size of the original array, the extra elements in the copy array are filled with zero values.\n * \n * @sample
samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n * \n\npublic actual fun
ByteArray.copyOfOf(newSize: Int): ByteArray {\n    require(newSize >= 0) { "<code>Invalid new array size: $newSize.</code>"
}\n    return fillFrom(this, ByteArray(newSize))\n}\n\n/**\n * Returns new array which is a copy of the original
array, resized to the given [newSize].\n * The copy is either truncated or padded at the end with zero values if
necessary.\n * \n * - If [newSize] is less than the size of the original array, the copy array is truncated to the
[newSize].\n * - If [newSize] is greater than the size of the original array, the extra elements in the copy array are
filled with zero values.\n * \n * @sample samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n
* \n\npublic actual fun ShortArray.copyOfOf(newSize: Int): ShortArray {\n    require(newSize >= 0) { "<code>Invalid new
array size: $newSize.</code>" }\n    return fillFrom(this, ShortArray(newSize))\n}\n\n/**\n * Returns new array which is a
copy of the original array, resized to the given [newSize].\n * The copy is either truncated or padded at the end with
zero values if necessary.\n * \n * - If [newSize] is less than the size of the original array, the copy array is truncated
to the [newSize].\n * - If [newSize] is greater than the size of the original array, the extra elements in the copy array
are filled with zero values.\n * \n * @sample
samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n * \n\npublic actual fun
IntArray.copyOfOf(newSize: Int): IntArray {\n    require(newSize >= 0) { "<code>Invalid new array size: $newSize.</code>" }\n
return fillFrom(this, IntArray(newSize))\n}\n\n/**\n * Returns new array which is a copy of the original array,
resized to the given [newSize].\n * The copy is either truncated or padded at the end with zero values if necessary.\n
* \n * - If [newSize] is less than the size of the original array, the copy array is truncated to the [newSize].\n * - If
[newSize] is greater than the size of the original array, the extra elements in the copy array are filled with zero
values.\n * \n * @sample samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n * \n\npublic actual
fun LongArray.copyOfOf(newSize: Int): LongArray {\n    require(newSize >= 0) { "<code>Invalid new array size:
$newSize.</code>" }\n    return withType("<code>LongArray</code>", arrayCopyResize(this, newSize, 0L))\n}\n\n/**\n * Returns new
array which is a copy of the original array, resized to the given [newSize].\n * The copy is either truncated or padded
at the end with zero values if necessary.\n * \n * - If [newSize] is less than the size of the original array, the copy
array is truncated to the [newSize].\n * - If [newSize] is greater than the size of the original array, the extra elements
in the copy array are filled with zero values.\n * \n * @sample
samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n * \n\npublic actual fun
FloatArray.copyOfOf(newSize: Int): FloatArray {\n    require(newSize >= 0) { "<code>Invalid new array size: $newSize.</code>"
}\n    return fillFrom(this, FloatArray(newSize))\n}\n\n/**\n * Returns new array which is a copy of the original
array, resized to the given [newSize].\n * The copy is either truncated or padded at the end with zero values if
necessary.\n * \n * - If [newSize] is less than the size of the original array, the copy array is truncated to the
[newSize].\n * - If [newSize] is greater than the size of the original array, the extra elements in the copy array are
filled with zero values.\n * \n * @sample samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n
* \n\npublic actual fun DoubleArray.copyOfOf(newSize: Int): DoubleArray {\n    require(newSize >= 0) { "<code>Invalid new
array size: $newSize.</code>" }\n    return fillFrom(this, DoubleArray(newSize))\n}\n\n/**\n * Returns new array which is
a copy of the original array, resized to the given [newSize].\n * The copy is either truncated or padded at the end
with `false` values if necessary.\n * \n * - If [newSize] is less than the size of the original array, the copy array is
truncated to the [newSize].\n * - If [newSize] is greater than the size of the original array, the extra elements in the
copy array are filled with `false` values.\n * \n * @sample

```

```

samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n *^\\npublic actual fun
BooleanArray.copyOf(newSize: Int): BooleanArray {\n  require(newSize >= 0) { \"Invalid new array size:
$newSize.\" }\n  return withType(\"BooleanArray\", arrayCopyResize(this, newSize, false))\n}\n\n/**\n * Returns
new array which is a copy of the original array, resized to the given [newSize].\n * The copy is either truncated or
padded at the end with null char (``\u0000``) values if necessary.\n * \n * - If [newSize] is less than the size of the
original array, the copy array is truncated to the [newSize].\n * - If [newSize] is greater than the size of the original
array, the extra elements in the copy array are filled with null char (``\u0000``) values.\n * \n * @sample
samples.collections.Arrays.CopyOfOperations.resizedPrimitiveCopyOf\n *^\\npublic actual fun
CharArray.copyOf(newSize: Int): CharArray {\n  require(newSize >= 0) { \"Invalid new array size: $newSize.\"
}\n  return withType(\"CharArray\", fillFrom(this, CharArray(newSize)))\n}\n\n/**\n * Returns new array which is
a copy of the original array, resized to the given [newSize].\n * The copy is either truncated or padded at the end
with `null` values if necessary.\n * \n * - If [newSize] is less than the size of the original array, the copy array is
truncated to the [newSize].\n * - If [newSize] is greater than the size of the original array, the extra elements in the
copy array are filled with `null` values.\n * \n * @sample
samples.collections.Arrays.CopyOfOperations.resizingCopyOf\n
*^\\n@Suppress(\"ACTUAL_WITHOUT_EXPECT\")\\npublic actual fun <T> Array<out T>.copyOf(newSize: Int):
Array<T?> {\n  require(newSize >= 0) { \"Invalid new array size: $newSize.\" }\n  return arrayCopyResize(this,
newSize, null)\n}\n\n/**\n * Returns a new array which is a copy of the specified range of the original array.\n * \n
* @param fromIndex the start of the range (inclusive) to copy.\n * @param toIndex the end of the range (exclusive)
to copy.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than
the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n
*^\\n@Suppress(\"ACTUAL_WITHOUT_EXPECT\")\\npublic actual fun <T> Array<out
T>.copyOfRange(fromIndex: Int, toIndex: Int): Array<T> {\n  AbstractList.checkRangeIndexes(fromIndex,
toIndex, size)\n  return this.asDynamic().slice(fromIndex, toIndex)\n}\n\n/**\n * Returns a new array which is a
copy of the specified range of the original array.\n * \n * @param fromIndex the start of the range (inclusive) to
copy.\n * @param toIndex the end of the range (exclusive) to copy.\n * \n * @throws IndexOutOfBoundsException
if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n * @throws
IllegalArgumentException if [fromIndex] is greater than [toIndex].\n *^\\npublic actual fun
ByteArray.copyOfRange(fromIndex: Int, toIndex: Int): ByteArray {\n
AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n  return this.asDynamic().slice(fromIndex,
toIndex)\n}\n\n/**\n * Returns a new array which is a copy of the specified range of the original array.\n * \n
* @param fromIndex the start of the range (inclusive) to copy.\n * @param toIndex the end of the range (exclusive) to
copy.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the
size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n *^\\npublic
actual fun ShortArray.copyOfRange(fromIndex: Int, toIndex: Int): ShortArray {\n
AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n  return this.asDynamic().slice(fromIndex,
toIndex)\n}\n\n/**\n * Returns a new array which is a copy of the specified range of the original array.\n * \n
* @param fromIndex the start of the range (inclusive) to copy.\n * @param toIndex the end of the range (exclusive) to
copy.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the
size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n *^\\npublic
actual fun IntArray.copyOfRange(fromIndex: Int, toIndex: Int): IntArray {\n
AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n  return this.asDynamic().slice(fromIndex,
toIndex)\n}\n\n/**\n * Returns a new array which is a copy of the specified range of the original array.\n * \n
* @param fromIndex the start of the range (inclusive) to copy.\n * @param toIndex the end of the range (exclusive) to
copy.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the
size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n *^\\npublic
actual fun LongArray.copyOfRange(fromIndex: Int, toIndex: Int): LongArray {\n
AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n  return withType(\"LongArray\",

```

```

this.asDynamic().slice(fromIndex, toIndex))\n}\n\n/**\n * Returns a new array which is a copy of the specified
range of the original array.\n * \n * @param fromIndex the start of the range (inclusive) to copy.\n * @param
toIndex the end of the range (exclusive) to copy.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is
less than zero or [toIndex] is greater than the size of this array.\n * @throws IllegalArgumentException if
[fromIndex] is greater than [toIndex].\n */\npublic actual fun FloatArray.copyOfRange(fromIndex: Int, toIndex: Int):
FloatArray {\n    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n    return
this.asDynamic().slice(fromIndex, toIndex)\n}\n\n/**\n * Returns a new array which is a copy of the specified range
of the original array.\n * \n * @param fromIndex the start of the range (inclusive) to copy.\n * @param toIndex the
end of the range (exclusive) to copy.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero
or [toIndex] is greater than the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater
than [toIndex].\n */\npublic actual fun DoubleArray.copyOfRange(fromIndex: Int, toIndex: Int): DoubleArray {\n
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n    return this.asDynamic().slice(fromIndex,
toIndex)\n}\n\n/**\n * Returns a new array which is a copy of the specified range of the original array.\n * \n *
@param fromIndex the start of the range (inclusive) to copy.\n * @param toIndex the end of the range (exclusive) to
copy.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the
size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n */\npublic
actual fun BooleanArray.copyOfRange(fromIndex: Int, toIndex: Int): BooleanArray {\n
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n    return withType(\n        "BooleanArray",
this.asDynamic().slice(fromIndex, toIndex))\n}\n\n/**\n * Returns a new array which is a copy of the specified
range of the original array.\n * \n * @param fromIndex the start of the range (inclusive) to copy.\n * @param
toIndex the end of the range (exclusive) to copy.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is
less than zero or [toIndex] is greater than the size of this array.\n * @throws IllegalArgumentException if
[fromIndex] is greater than [toIndex].\n */\npublic actual fun CharArray.copyOfRange(fromIndex: Int, toIndex: Int):
CharArray {\n    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n    return withType(\n        "CharArray",
this.asDynamic().slice(fromIndex, toIndex))\n}\n\n/**\n * Fills this array or its subrange with the specified
[element] value.\n * \n * @param fromIndex the start of the range (inclusive) to fill, 0 by default.\n * @param
toIndex the end of the range (exclusive) to fill, size of this array by default.\n * \n * @throws
IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n *
@throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n */\n\n*\n@SinceKotlin("1.3")\n@Suppress(\n    "ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic
actual fun <T> Array<T>.fill(element: T, fromIndex: Int = 0, toIndex: Int = size): Unit {\n
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n    this.asDynamic().fill(element, fromIndex,
toIndex);\n}\n\n/**\n * Fills this array or its subrange with the specified [element] value.\n * \n * @param
fromIndex the start of the range (inclusive) to fill, 0 by default.\n * @param toIndex the end of the range (exclusive)
to fill, size of this array by default.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero
or [toIndex] is greater than the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater
than [toIndex].\n */\n\n*\n@SinceKotlin("1.3")\n@Suppress(\n    "ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic
actual fun ByteArray.fill(element: Byte, fromIndex: Int = 0, toIndex: Int = size): Unit {\n
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n    this.asDynamic().fill(element, fromIndex,
toIndex);\n}\n\n/**\n * Fills this array or its subrange with the specified [element] value.\n * \n * @param
fromIndex the start of the range (inclusive) to fill, 0 by default.\n * @param toIndex the end of the range (exclusive)
to fill, size of this array by default.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero
or [toIndex] is greater than the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater
than [toIndex].\n */\n\n*\n@SinceKotlin("1.3")\n@Suppress(\n    "ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic
actual fun ShortArray.fill(element: Short, fromIndex: Int = 0, toIndex: Int = size): Unit {\n
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n    this.asDynamic().fill(element, fromIndex,

```

```

toIndex);}

/**
 * Fills this array or its subrange with the specified [element] value.
 * @param fromIndex the start of the range (inclusive) to fill, 0 by default.
 * @param toIndex the end of the range (exclusive) to fill, size of this array by default.
 * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.3")
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual fun IntArray.fill(element: Int, fromIndex: Int = 0, toIndex: Int = size): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)
    this.asDynamic().fill(element, fromIndex, toIndex);
}

/**
 * Fills this array or its subrange with the specified [element] value.
 * @param fromIndex the start of the range (inclusive) to fill, 0 by default.
 * @param toIndex the end of the range (exclusive) to fill, size of this array by default.
 * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.3")
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual fun LongArray.fill(element: Long, fromIndex: Int = 0, toIndex: Int = size): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)
    this.asDynamic().fill(element, fromIndex, toIndex);
}

/**
 * Fills this array or its subrange with the specified [element] value.
 * @param fromIndex the start of the range (inclusive) to fill, 0 by default.
 * @param toIndex the end of the range (exclusive) to fill, size of this array by default.
 * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.3")
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual fun FloatArray.fill(element: Float, fromIndex: Int = 0, toIndex: Int = size): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)
    this.asDynamic().fill(element, fromIndex, toIndex);
}

/**
 * Fills this array or its subrange with the specified [element] value.
 * @param fromIndex the start of the range (inclusive) to fill, 0 by default.
 * @param toIndex the end of the range (exclusive) to fill, size of this array by default.
 * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.3")
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual fun DoubleArray.fill(element: Double, fromIndex: Int = 0, toIndex: Int = size): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)
    this.asDynamic().fill(element, fromIndex, toIndex);
}

/**
 * Fills this array or its subrange with the specified [element] value.
 * @param fromIndex the start of the range (inclusive) to fill, 0 by default.
 * @param toIndex the end of the range (exclusive) to fill, size of this array by default.
 * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.3")
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual fun BooleanArray.fill(element: Boolean, fromIndex: Int = 0, toIndex: Int = size): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)
    this.asDynamic().fill(element, fromIndex, toIndex);
}

/**
 * Fills this array or its subrange with the specified [element] value.
 * @param fromIndex the start of the range (inclusive) to fill, 0 by default.
 * @param toIndex the end of the range (exclusive) to fill, size of this array by default.
 * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.
 * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].
 */
@SinceKotlin("1.3")
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual fun CharArray.fill(element: Char, fromIndex: Int = 0, toIndex: Int = size): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)
    this.asDynamic().fill(element, fromIndex,

```

toIndex);

`<T> Array<out T>.plus(element: T): Array<T>` {\n return this.asDynamic().concat(arrayOf(element))}

`<T> Array<out T>.plus(element: T): Array<T>` {\n return this.asDynamic().concat(arrayOf(element))}

`ByteArray.plus(element: Byte): ByteArray` {\n return plus(byteArrayOf(element))}

`ShortArray.plus(element: Short): ShortArray` {\n return plus(shortArrayOf(element))}

`IntArray.plus(element: Int): IntArray` {\n return plus(intArrayOf(element))}

`LongArray.plus(element: Long): LongArray` {\n return plus(longArrayOf(element))}

`FloatArray.plus(element: Float): FloatArray` {\n return plus(floatArrayOf(element))}

`DoubleArray.plus(element: Double): DoubleArray` {\n return plus(doubleArrayOf(element))}

`BooleanArray.plus(element: Boolean): BooleanArray` {\n return plus(booleanArrayOf(element))}

`CharArray.plus(element: Char): CharArray` {\n return plus(charArrayOf(element))}

`<T> Array<out T>.plus(elements: Collection<T>): Array<T>` {\n return arrayPlusCollection(this, elements)}

`ByteArray.plus(elements: Collection<Byte>): ByteArray` {\n return fillFromCollection(this.copyOf(size + elements.size), this.size, elements)}

`ShortArray.plus(elements: Collection<Short>): ShortArray` {\n return fillFromCollection(this.copyOf(size + elements.size), this.size, elements)}

`IntArray.plus(elements: Collection<Int>): IntArray` {\n return fillFromCollection(this.copyOf(size + elements.size), this.size, elements)}

`LongArray.plus(elements: Collection<Long>): LongArray` {\n return arrayPlusCollection(this, elements)}

`FloatArray.plus(elements: Collection<Float>): FloatArray` {\n return fillFromCollection(this.copyOf(size + elements.size), this.size, elements)}

`DoubleArray.plus(elements: Collection<Double>): DoubleArray` {\n return fillFromCollection(this.copyOf(size + elements.size), this.size, elements)}

`BooleanArray.plus(elements: Collection<Boolean>): BooleanArray` {\n return arrayPlusCollection(this, elements)}

`CharArray.plus(elements: Collection<Char>): CharArray` {\n return fillFromCollection(this.copyOf(size + elements.size), this.size, elements)}

```

Collection<Char>): CharArray {\n  return fillFromCollection(this.copyOf(size + elements.size), this.size,
elements)\n}\n\n/**\n * Returns an array containing all elements of the original array and then all elements of the
given [elements] array.\n */\n@Suppress("ACTUAL_WITHOUT_EXPECT",
"NOTHING_TO_INLINE")\npublic actual inline operator fun <T> Array<out T>.plus(elements: Array<out T>):
Array<T> {\n  return this.asDynamic().concat(elements)\n}\n\n/**\n * Returns an array containing all elements of
the original array and then all elements of the given [elements] array.\n
*/\n@Suppress("NOTHING_TO_INLINE")\npublic actual inline operator fun ByteArray.plus(elements:
ByteArray): ByteArray {\n  return primitiveArrayConcat(this, elements)\n}\n\n/**\n * Returns an array containing
all elements of the original array and then all elements of the given [elements] array.\n
*/\n@Suppress("NOTHING_TO_INLINE")\npublic actual inline operator fun ShortArray.plus(elements:
ShortArray): ShortArray {\n  return primitiveArrayConcat(this, elements)\n}\n\n/**\n * Returns an array
containing all elements of the original array and then all elements of the given [elements] array.\n
*/\n@Suppress("NOTHING_TO_INLINE")\npublic actual inline operator fun IntArray.plus(elements: IntArray):
IntArray {\n  return primitiveArrayConcat(this, elements)\n}\n\n/**\n * Returns an array containing all elements of
the original array and then all elements of the given [elements] array.\n
*/\n@Suppress("NOTHING_TO_INLINE")\npublic actual inline operator fun LongArray.plus(elements:
LongArray): LongArray {\n  return primitiveArrayConcat(this, elements)\n}\n\n/**\n * Returns an array
containing all elements of the original array and then all elements of the given [elements] array.\n
*/\n@Suppress("NOTHING_TO_INLINE")\npublic actual inline operator fun FloatArray.plus(elements:
FloatArray): FloatArray {\n  return primitiveArrayConcat(this, elements)\n}\n\n/**\n * Returns an array containing
all elements of the original array and then all elements of the given [elements] array.\n
*/\n@Suppress("NOTHING_TO_INLINE")\npublic actual inline operator fun DoubleArray.plus(elements:
DoubleArray): DoubleArray {\n  return primitiveArrayConcat(this, elements)\n}\n\n/**\n * Returns an array
containing all elements of the original array and then all elements of the given [elements] array.\n
*/\n@Suppress("NOTHING_TO_INLINE")\npublic actual inline operator fun BooleanArray.plus(elements:
BooleanArray): BooleanArray {\n  return primitiveArrayConcat(this, elements)\n}\n\n/**\n * Returns an array
containing all elements of the original array and then all elements of the given [elements] array.\n
*/\n@Suppress("NOTHING_TO_INLINE")\npublic actual inline operator fun CharArray.plus(elements:
CharArray): CharArray {\n  return primitiveArrayConcat(this, elements)\n}\n\n/**\n * Returns an array containing
all elements of the original array and then the given [element].\n
*/\n@Suppress("ACTUAL_WITHOUT_EXPECT", "NOTHING_TO_INLINE")\npublic actual inline fun <T>
Array<out T>.plusElement(element: T): Array<T> {\n  return
this.asDynamic().concat(arrayOf(element))\n}\n\n/**\n * Sorts the array in-place.\n */\n * @sample
samples.collections.Arrays.Sorting.sortArray\n */\n@library("primitiveArraySort")\npublic actual fun
IntArray.sort(): Unit {\n  definedExternally\n}\n\n/**\n * Sorts the array in-place.\n */\n * @sample
samples.collections.Arrays.Sorting.sortArray\n */\npublic actual fun LongArray.sort(): Unit {\n
@Suppress("DEPRECATION")\n  if (size > 1) sort { a: Long, b: Long -> a.compareTo(b) }\n}\n\n/**\n * Sorts
the array in-place.\n */\n * @sample samples.collections.Arrays.Sorting.sortArray\n
*/\n@library("primitiveArraySort")\npublic actual fun ByteArray.sort(): Unit {\n  definedExternally\n}\n\n/**\n
* Sorts the array in-place.\n */\n * @sample samples.collections.Arrays.Sorting.sortArray\n
*/\n@library("primitiveArraySort")\npublic actual fun ShortArray.sort(): Unit {\n  definedExternally\n}\n\n/**\n
* Sorts the array in-place.\n */\n * @sample samples.collections.Arrays.Sorting.sortArray\n
*/\n@library("primitiveArraySort")\npublic actual fun DoubleArray.sort(): Unit {\n
definedExternally\n}\n\n/**\n * Sorts the array in-place.\n */\n * @sample
samples.collections.Arrays.Sorting.sortArray\n */\n@library("primitiveArraySort")\npublic actual fun
FloatArray.sort(): Unit {\n  definedExternally\n}\n\n/**\n * Sorts the array in-place.\n */\n * @sample
samples.collections.Arrays.Sorting.sortArray\n */\n@library("primitiveArraySort")\npublic actual fun
CharArray.sort(): Unit {\n  definedExternally\n}\n\n/**\n * Sorts the array in-place according to the natural order

```

of its elements. The sort is `_stable_`. It means that equal elements preserve their order relative to each other after sorting.

```

@sample samples.collections.Arrays.Sorting.sortArrayOfComparable
public actual fun
<T : Comparable<T>> Array<out T>.sort(): Unit {
    if (size > 1) sortArray(this)
}

Sorts the array in-place according to the order specified by the given [comparison] function. The sort is _stable_. It means that equal elements preserve their order relative to each other after sorting.

@Deprecated("Use sortWith instead",
ReplaceWith("this.sortWith(Comparator(comparison))"))
@DeprecatedSinceKotlin(warningSince =
"1.6")
public fun <T> Array<out T>.sort(comparison: (a: T, b: T) -> Int): Unit {
    if (size > 1)
        sortArrayWith(this, comparison)
}

Sorts a range in the array in-place. The sort is _stable_. It means that equal elements preserve their order relative to each other after sorting.

@param fromIndex the start of the range (inclusive) to sort, 0 by default.
@param toIndex the end of the range (exclusive) to sort, size of this array by default.
@throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.
@throws IllegalArgumentException if [fromIndex] is greater than [toIndex].

@sample samples.collections.Arrays.Sorting.sortRangeOfArrayOfComparable
@SinceKotlin("1.4")
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual fun <T : Comparable<T>> Array<out T>.sort(fromIndex: Int = 0, toIndex: Int = size): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)
    sortArrayWith(this, fromIndex, toIndex,
        naturalOrder())
}

Sorts a range in the array in-place.

@param fromIndex the start of the range (inclusive) to sort, 0 by default.
@param toIndex the end of the range (exclusive) to sort, size of this array by default.
@throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.
@throws IllegalArgumentException if [fromIndex] is greater than [toIndex].

@sample samples.collections.Arrays.Sorting.sortRangeOfArray
@SinceKotlin("1.4")
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual fun ByteArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)
    val subarray =
        this.asDynamic().subarray(fromIndex, toIndex).unsafeCast<ByteArray>()
    subarray.sort()
}

Sorts a range in the array in-place.

@param fromIndex the start of the range (inclusive) to sort, 0 by default.
@param toIndex the end of the range (exclusive) to sort, size of this array by default.
@throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.
@throws IllegalArgumentException if [fromIndex] is greater than [toIndex].

@sample samples.collections.Arrays.Sorting.sortRangeOfArray
@SinceKotlin("1.4")
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual fun ShortArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)
    val subarray =
        this.asDynamic().subarray(fromIndex, toIndex).unsafeCast<ShortArray>()
    subarray.sort()
}

Sorts a range in the array in-place.

@param fromIndex the start of the range (inclusive) to sort, 0 by default.
@param toIndex the end of the range (exclusive) to sort, size of this array by default.
@throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.
@throws IllegalArgumentException if [fromIndex] is greater than [toIndex].

@sample samples.collections.Arrays.Sorting.sortRangeOfArray
@SinceKotlin("1.4")
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual fun IntArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit {
    AbstractList.checkRangeIndexes(fromIndex, toIndex, size)
    val subarray =
        this.asDynamic().subarray(fromIndex, toIndex).unsafeCast<IntArray>()
    subarray.sort()
}

Sorts a range in the array in-place.

@param fromIndex the start of the range (inclusive) to sort, 0 by default.
@param toIndex the end of the range (exclusive) to sort, size of this array by default.
@throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.
@throws IllegalArgumentException if [fromIndex] is greater than [toIndex].

@sample samples.collections.Arrays.Sorting.sortRangeOfArray

```

```

*\n@SinceKotlin("1.4")\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic
actual fun LongArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit {\n
AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n    sortArrayWith(this.unsafeCast<Array<Long>>(),
fromIndex, toIndex, naturalOrder())\n}\n\n/**\n * Sorts a range in the array in-place.\n * \n * @param fromIndex
the start of the range (inclusive) to sort, 0 by default.\n * @param toIndex the end of the range (exclusive) to sort,
size of this array by default.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or
[toIndex] is greater than the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than
[toIndex].\n * \n * @sample samples.collections.Arrays.Sorting.sortRangeOfArray\n
*\n@SinceKotlin("1.4")\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic
actual fun FloatArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit {\n
AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n    val subarray =
this.asDynamic().subarray(fromIndex, toIndex).unsafeCast<FloatArray>()\n    subarray.sort()\n}\n\n/**\n * Sorts a
range in the array in-place.\n * \n * @param fromIndex the start of the range (inclusive) to sort, 0 by default.\n *
@param toIndex the end of the range (exclusive) to sort, size of this array by default.\n * \n * @throws
IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n *
@throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n * \n * @sample
samples.collections.Arrays.Sorting.sortRangeOfArray\n
*\n@SinceKotlin("1.4")\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic
actual fun DoubleArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit {\n
AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n    val subarray =
this.asDynamic().subarray(fromIndex, toIndex).unsafeCast<DoubleArray>()\n    subarray.sort()\n}\n\n/**\n * Sorts
a range in the array in-place.\n * \n * @param fromIndex the start of the range (inclusive) to sort, 0 by default.\n *
@param toIndex the end of the range (exclusive) to sort, size of this array by default.\n * \n * @throws
IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n *
@throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n * \n * @sample
samples.collections.Arrays.Sorting.sortRangeOfArray\n
*\n@SinceKotlin("1.4")\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic
actual fun CharArray.sort(fromIndex: Int = 0, toIndex: Int = size): Unit {\n
AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n    val subarray =
this.asDynamic().subarray(fromIndex, toIndex).unsafeCast<CharArray>()\n    subarray.sort()\n}\n\n/**\n * Sorts the
array in-place according to the order specified by the given [comparison] function.\n * \n * @Deprecated("Use other
sorting functions from the Standard Library")\n * @DeprecatedSinceKotlin(warningSince =
"1.6")\n * @kotlin.internal.InlineOnly\n * public inline fun ByteArray.sort(noinline comparison: (a: Byte, b: Byte) ->
Int): Unit {\n    asDynamic().sort(comparison)\n}\n\n/**\n * Sorts the array in-place according to the order specified
by the given [comparison] function.\n * \n * @Deprecated("Use other sorting functions from the Standard
Library")\n * @DeprecatedSinceKotlin(warningSince = "1.6")\n * @kotlin.internal.InlineOnly\n * public inline fun
ShortArray.sort(noinline comparison: (a: Short, b: Short) -> Int): Unit {\n
asDynamic().sort(comparison)\n}\n\n/**\n * Sorts the array in-place according to the order specified by the given
[comparison] function.\n * \n * @Deprecated("Use other sorting functions from the Standard
Library")\n * @DeprecatedSinceKotlin(warningSince = "1.6")\n * @kotlin.internal.InlineOnly\n * public inline fun
IntArray.sort(noinline comparison: (a: Int, b: Int) -> Int): Unit {\n    asDynamic().sort(comparison)\n}\n\n/**\n *
Sorts the array in-place according to the order specified by the given [comparison] function.\n * \n * @Deprecated("Use other sorting functions from the Standard
Library")\n * @DeprecatedSinceKotlin(warningSince = "1.6")\n * @kotlin.internal.InlineOnly\n * public inline fun
LongArray.sort(noinline comparison: (a: Long, b: Long) -> Int): Unit {\n
asDynamic().sort(comparison)\n}\n\n/**\n * Sorts the array in-place according to the order specified by the given
[comparison] function.\n * \n * @Deprecated("Use other sorting functions from the Standard
Library")\n * @DeprecatedSinceKotlin(warningSince = "1.6")\n * @kotlin.internal.InlineOnly\n * public inline fun

```

```

FloatArray.sort(noinline comparison: (a: Float, b: Float) -> Int): Unit {
asDynamic().sort(comparison)}\n\n/**\n * Sorts the array in-place according to the order specified by the given
[comparison] function.\n */\n@Deprecated("Use other sorting functions from the Standard
Library")\n@DeprecatedSinceKotlin(warningSince = "1.6")\n@kotlin.internal.InlineOnly\npublic inline fun
DoubleArray.sort(noinline comparison: (a: Double, b: Double) -> Int): Unit {
asDynamic().sort(comparison)}\n\n/**\n * Sorts the array in-place according to the order specified by the given
[comparison] function.\n */\n@Deprecated("Use other sorting functions from the Standard
Library")\n@DeprecatedSinceKotlin(warningSince = "1.6")\n@kotlin.internal.InlineOnly\npublic inline fun
CharArray.sort(noinline comparison: (a: Char, b: Char) -> Int): Unit {
asDynamic().sort(comparison)}\n\n/**\n * The sort is _stable_. It means that equal elements preserve their order relative to each other
after sorting.\n */\npublic actual fun <T> Array<out T>.sortWith(comparator: Comparator<in T>): Unit {
if (size > 1) sortArrayWith(this, comparator)}\n\n/**\n * Sorts a range in the array in-place with the given
[comparator].\n */\n * The sort is _stable_. It means that equal elements preserve their order relative to each other
after sorting.\n */\n * @param fromIndex the start of the range (inclusive) to sort, 0 by default.\n * @param toIndex
the end of the range (exclusive) to sort, size of this array by default.\n * \n * @throws IndexOutOfBoundsException
if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n * @throws
IllegalArgumentExcepion if [fromIndex] is greater than [toIndex].\n
*/\n@SinceKotlin("1.4")\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic
actual fun <T> Array<out T>.sortWith(comparator: Comparator<in T>, fromIndex: Int = 0, toIndex: Int = size):
Unit {
AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n sortArrayWith(this, fromIndex, toIndex,
comparator)}\n\n/**\n * Returns a *typed* object array containing all of the elements of this primitive array.\n
*/\npublic actual fun ByteArray.toTypedArray(): Array<Byte> {
return js("[]").slice.call(this)}\n\n/**\n * Returns a *typed* object array containing all of the elements of this primitive array.\n
*/\npublic actual fun ShortArray.toTypedArray(): Array<Short> {
return js("[]").slice.call(this)}\n\n/**\n * Returns a *typed*
object array containing all of the elements of this primitive array.\n */\npublic actual fun IntArray.toTypedArray():
Array<Int> {
return js("[]").slice.call(this)}\n\n/**\n * Returns a *typed* object array containing all of the elements of this primitive
array.\n */\npublic actual fun LongArray.toTypedArray(): Array<Long> {
return js("[]").slice.call(this)}\n\n/**\n * Returns a *typed* object array containing all of the elements of this primitive
array.\n */\npublic actual fun FloatArray.toTypedArray(): Array<Float> {
return js("[]").slice.call(this)}\n\n/**\n * Returns a *typed* object array containing all of the elements of this primitive
array.\n */\npublic actual fun DoubleArray.toTypedArray(): Array<Double> {
return js("[]").slice.call(this)}\n\n/**\n * Returns a *typed* object array containing all of the elements of this primitive
array.\n */\npublic actual fun BooleanArray.toTypedArray(): Array<Boolean> {
return js("[]").slice.call(this)}\n\n/**\n * Returns a *typed* object array containing all of the elements of this primitive
array.\n */\npublic actual fun CharArray.toTypedArray(): Array<Char> {
return Array(size) { index ->
this[index] }}\n\n", "/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n
*/\n@file:kotlin.jvm.JvmName("ComparisonsKt")\n@file:kotlin.jvm.JvmMultifileClass\npackage
kotlin.comparisons\n\n/**\n * Compares two values using the specified functions [selectors] to calculate the result
of the comparison.\n * The functions are called sequentially, receive the given values [a] and [b] and return
[Comparable]\n * objects. As soon as the [Comparable] instances returned by a function for [a] and [b] values do
not\n * compare as equal, the result of that comparison is returned.\n */\n * @sample
samples.comparisons.Comparisons.compareValuesByWithSelectors\n */\npublic fun <T> compareValuesBy(a: T,
b: T, vararg selectors: (T) -> Comparable<*>?): Int {
require(selectors.size > 0)\n return
compareValuesByImpl(a, b, selectors)}\n\nprivate fun <T> compareValuesByImpl(a: T, b: T, selectors:
Array<out (T) -> Comparable<*>?): Int {
for (fn in selectors) {
val v1 = fn(a)\n val v2 = fn(b)\n

```

```

val diff = compareValues(v1, v2)\n    if (diff != 0) return diff\n    } \n    return 0\n}\n\n/**\n * Compares two
values using the specified [selector] function to calculate the result of the comparison.\n * The function is applied to
the given values [a] and [b] and return [Comparable] objects.\n * The result of comparison of these [Comparable]
instances is returned.\n *\n * @sample samples.comparisons.Comparisons.compareValuesByWithSingleSelector\n
*\n@kotlin.internal.InlineOnly\npublic inline fun <T> compareValuesBy(a: T, b: T, selector: (T) ->
Comparable<*>?): Int {\n    return compareValues(selector(a), selector(b))\n}\n\n/**\n * Compares two values
using the specified [selector] function to calculate the result of the comparison.\n * The function is applied to the
given values [a] and [b] and return objects of type K which are then being\n * compared with the given
[comparator].\n *\n * @sample samples.comparisons.Comparisons.compareValuesByWithComparator\n
*\n@kotlin.internal.InlineOnly\npublic inline fun <T, K> compareValuesBy(a: T, b: T, comparator: Comparator<in
K>, selector: (T) -> K): Int {\n    return comparator.compare(selector(a), selector(b))\n}\n\n//// Not so useful without
type inference for receiver of expression\n//// compareValuesWith(v1, v2, compareBy { it.prop1 }
thenByDescending { it.prop2 })\n\n/**\n * Compares two values using the specified [comparator].\n\n
*\n@Suppress("NOTHING_TO_INLINE")\npublic inline fun <T> compareValuesWith(a: T, b: T, comparator:
Comparator<T>): Int = comparator.compare(a, b)\n\n/**\n * Compares two nullable [Comparable] values. Null
is considered less than any value.\n *\n * @sample samples.comparisons.Comparisons.compareValues\n *\npublic
fun <T : Comparable<*>> compareValues(a: T?, b: T?): Int {\n    if (a === b) return 0\n    if (a == null) return -1\n
if (b == null) return 1\n    @Suppress("UNCHECKED_CAST")\n    return (a as
Comparable<Any>).compareTo(b)\n}\n\n/**\n * Creates a comparator using the sequence of functions to calculate a
result of comparison.\n * The functions are called sequentially, receive the given values `a` and `b` and return
[Comparable]\n * objects. As soon as the [Comparable] instances returned by a function for `a` and `b` values do
not\n * compare as equal, the result of that comparison is returned from the [Comparator].\n *\n * @sample
samples.comparisons.Comparisons.compareByWithSelectors\n *\npublic fun <T> compareBy(vararg selectors: (T)
-> Comparable<*>?): Comparator<T> {\n    require(selectors.size > 0)\n    return Comparator { a, b ->
compareValuesByImpl(a, b, selectors) }\n}\n\n/**\n * Creates a comparator using the function to transform value
to a [Comparable] instance for comparison.\n *\n * @sample
samples.comparisons.Comparisons.compareByWithSingleSelector\n *\n@kotlin.internal.InlineOnly\npublic inline
fun <T> compareBy(crossinline selector: (T) -> Comparable<*>?): Comparator<T> =\n    Comparator { a, b ->
compareValuesBy(a, b, selector) }\n\n/**\n * Creates a comparator using the [selector] function to transform values
being compared and then applying\n * the specified [comparator] to compare transformed values.\n *\n * @sample
samples.comparisons.Comparisons.compareByWithComparator\n *\n@kotlin.internal.InlineOnly\npublic inline
fun <T, K> compareBy(comparator: Comparator<in K>, crossinline selector: (T) -> K): Comparator<T> =\n    Comparator { a, b -> compareValuesBy(a, b, comparator, selector) }\n\n/**\n * Creates a descending comparator
using the function to transform value to a [Comparable] instance for comparison.\n *\n * @sample
samples.comparisons.Comparisons.compareByDescendingWithSingleSelector\n
*\n@kotlin.internal.InlineOnly\npublic inline fun <T> compareByDescending(crossinline selector: (T) ->
Comparable<*>?): Comparator<T> =\n    Comparator { a, b -> compareValuesBy(b, a, selector) }\n\n/**\n *
Creates a descending comparator using the [selector] function to transform values being compared and then
applying\n * the specified [comparator] to compare transformed values.\n *\n * Note that an order of [comparator] is
reversed by this wrapper.\n *\n * @sample
samples.comparisons.Comparisons.compareByDescendingWithComparator\n
*\n@kotlin.internal.InlineOnly\npublic inline fun <T, K> compareByDescending(comparator: Comparator<in K>,
crossinline selector: (T) -> K): Comparator<T> =\n    Comparator { a, b -> compareValuesBy(b, a, comparator,
selector) }\n\n/**\n * Creates a comparator comparing values after the primary comparator defined them equal. It
uses\n * the function to transform value to a [Comparable] instance for comparison.\n *\n * @sample
samples.comparisons.Comparisons.thenBy\n *\n@kotlin.internal.InlineOnly\npublic inline fun <T>
Comparator<T>.thenBy(crossinline selector: (T) -> Comparable<*>?): Comparator<T> =\n    Comparator { a, b -
>\n        val previousCompare = this@thenBy.compare(a, b)\n        if (previousCompare != 0) previousCompare else

```

```

compareValuesBy(a, b, selector)\n } \n\n/**\n * Creates a comparator comparing values after the primary
comparator defined them equal. It uses\n * the [selector] function to transform values and then compares them with
the given [comparator].\n *\n * @sample samples.comparisons.Comparisons.thenByWithComparator\n
*\n@kotlin.internal.InlineOnly\npublic inline fun <T, K> Comparator<T>.thenBy(comparator: Comparator<in K>,
crossinline selector: (T) -> K): Comparator<T> =\n    Comparator { a, b ->\n        val previousCompare =
this@thenBy.compare(a, b)\n        if (previousCompare != 0) previousCompare else compareValuesBy(a, b,
comparator, selector)\n    } \n\n/**\n * Creates a descending comparator using the primary comparator and\n * the
function to transform value to a [Comparable] instance for comparison.\n *\n * @sample
samples.comparisons.Comparisons.thenByDescending\n *\n@kotlin.internal.InlineOnly\npublic inline fun <T>
Comparator<T>.thenByDescending(crossinline selector: (T) -> Comparable<*>?): Comparator<T> =\n
    Comparator { a, b ->\n        val previousCompare = this@thenByDescending.compare(a, b)\n        if
(previousCompare != 0) previousCompare else compareValuesBy(b, a, selector)\n    } \n\n/**\n * Creates a
descending comparator comparing values after the primary comparator defined them equal. It uses\n * the [selector]
function to transform values and then compares them with the given [comparator].\n *\n * @sample
samples.comparisons.Comparisons.thenByDescendingWithComparator\n *\n@kotlin.internal.InlineOnly\npublic
inline fun <T, K> Comparator<T>.thenByDescending(comparator: Comparator<in K>, crossinline selector: (T) ->
K): Comparator<T> =\n    Comparator { a, b ->\n        val previousCompare = this@thenByDescending.compare(a,
b)\n        if (previousCompare != 0) previousCompare else compareValuesBy(b, a, comparator, selector)\n    }
\n\n/**\n * Creates a comparator using the primary comparator and function to calculate a result of comparison.\n
*\n * @sample samples.comparisons.Comparisons.thenComparator\n *\n@kotlin.internal.InlineOnly\npublic inline
fun <T> Comparator<T>.thenComparator(crossinline comparison: (a: T, b: T) -> Int): Comparator<T> =\n
    Comparator { a, b ->\n        val previousCompare = this@thenComparator.compare(a, b)\n        if (previousCompare
!= 0) previousCompare else comparison(a, b)\n    } \n\n/**\n * Combines this comparator and the given [comparator]
such that the latter is applied only\n * when the former considered values equal.\n *\n * @sample
samples.comparisons.Comparisons.then\n *\npublic infix fun <T> Comparator<T>.then(comparator:
Comparator<in T>): Comparator<T> =\n    Comparator { a, b ->\n        val previousCompare =
this@then.compare(a, b)\n        if (previousCompare != 0) previousCompare else comparator.compare(a, b)\n    }
\n\n/**\n * Combines this comparator and the given [comparator] such that the latter is applied only\n * when the
former considered values equal.\n *\n * @sample samples.comparisons.Comparisons.thenDescending\n *\npublic
infix fun <T> Comparator<T>.thenDescending(comparator: Comparator<in T>): Comparator<T> =\n
    Comparator<T> { a, b ->\n        val previousCompare = this@thenDescending.compare(a, b)\n        if
(previousCompare != 0) previousCompare else comparator.compare(b, a)\n    } \n\n// Not so useful without type
inference for receiver of expression\n\n/**\n * Extends the given [comparator] of non-nullable values to a comparator
of nullable values\n * considering `null` value less than any other value.\n *\n * @sample
samples.comparisons.Comparisons.nullsFirstLastWithComparator\n *\npublic fun <T : Any>
nullsFirst(comparator: Comparator<in T>): Comparator<T?> =\n    Comparator { a, b ->\n        when {\n            a
=== b -> 0\n            a == null -> -1\n            b == null -> 1\n            else -> comparator.compare(a, b)\n        }\n    }
\n\n/**\n * Provides a comparator of nullable [Comparable] values\n * considering `null` value less than any other
value.\n *\n * @sample samples.comparisons.Comparisons.nullsFirstLastComparator\n
*\n@kotlin.internal.InlineOnly\npublic inline fun <T : Comparable<T>> nullsFirst(): Comparator<T?> =
nullsFirst(naturalOrder())\n\n/**\n * Extends the given [comparator] of non-nullable values to a comparator of
nullable values\n * considering `null` value greater than any other value.\n *\n * @sample
samples.comparisons.Comparisons.nullsFirstLastWithComparator\n *\npublic fun <T : Any>
nullsLast(comparator: Comparator<in T>): Comparator<T?> =\n    Comparator { a, b ->\n        when {\n            a
=== b -> 0\n            a == null -> 1\n            b == null -> -1\n            else -> comparator.compare(a, b)\n        }\n    }
\n\n/**\n * Provides a comparator of nullable [Comparable] values\n * considering `null` value greater than any
other value.\n *\n * @sample samples.comparisons.Comparisons.nullsFirstLastComparator\n
*\n@kotlin.internal.InlineOnly\npublic inline fun <T : Comparable<T>> nullsLast(): Comparator<T?> =

```

```

nullsLast(naturalOrder())\n\n/**\n * Returns a comparator that compares [Comparable] objects in natural order.\n *\n * @sample samples.comparisons.Comparisons.naturalOrderComparator\n */\npublic fun <T : Comparable<T>>\n naturalOrder(): Comparator<T> = @Suppress("UNCHECKED_CAST") (NaturalOrderComparator as\n Comparator<T>)\n\n/**\n * Returns a comparator that compares [Comparable] objects in reversed natural order.\n *\n * @sample samples.comparisons.Comparisons.nullsFirstLastWithComparator\n */\npublic fun <T :\n Comparable<T>> reverseOrder(): Comparator<T> = @Suppress("UNCHECKED_CAST")\n (ReverseOrderComparator as Comparator<T>)\n\n/**\n * Returns a comparator that imposes the reverse ordering\n of this comparator.\n *\n * @sample samples.comparisons.Comparisons.reversed\n */\n@Suppress("EXTENSION_SHADOWED_BY_MEMBER")\npublic fun <T> Comparator<T>.reversed():\n Comparator<T> = when (this) {\n is ReversedComparator -> this.comparator\n NaturalOrderComparator ->\n @Suppress("UNCHECKED_CAST") (ReverseOrderComparator as Comparator<T>)\n ReverseOrderComparator -> @Suppress("UNCHECKED_CAST") (NaturalOrderComparator as\n Comparator<T>)\n else -> ReversedComparator(this)\n }\n\nprivate class ReversedComparator<T>(public val\n comparator: Comparator<T>) : Comparator<T> {\n override fun compare(a: T, b: T): Int = comparator.compare(b,\n a)\n @Suppress("VIRTUAL_MEMBER_HIDDEN")\n fun reversed(): Comparator<T> =\n comparator\n }\n\nprivate object NaturalOrderComparator : Comparator<Comparable<Any>> {\n override fun\n compare(a: Comparable<Any>, b: Comparable<Any>): Int = a.compareTo(b)\n }\n\n@Suppress("VIRTUAL_MEMBER_HIDDEN")\n fun reversed(): Comparator<Comparable<Any>> =\n ReverseOrderComparator\n }\n\nprivate object ReverseOrderComparator : Comparator<Comparable<Any>> {\n override fun\n compare(a: Comparable<Any>, b: Comparable<Any>): Int = b.compareTo(a)\n }\n\n@Suppress("VIRTUAL_MEMBER_HIDDEN")\n fun reversed(): Comparator<Comparable<Any>> =\n NaturalOrderComparator\n }\n\n"/**\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language\n contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the\n license/LICENSE.txt file.\n */\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("StandardKt")\npackage kotlin\n\nimport\n kotlin.contracts.*\n\n/**\n * An exception is thrown to indicate that a method body remains to be implemented.\n */\npublic class NotImplementedError(message: String = "An operation is not implemented.") :\n Error(message)\n\n/**\n * Always throws [NotImplementedError] stating that operation is not implemented.\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun TODO(): Nothing = throw NotImplementedError()\n\n/**\n * Always throws [NotImplementedError] stating that operation is not implemented.\n *\n * @param reason a string\n explaining why the implementation is missing.\n */\n@kotlin.internal.InlineOnly\npublic inline fun TODO(reason:\n String): Nothing = throw NotImplementedError("An operation is not implemented: $reason")\n\n/**\n * Calls\n the specified function [block] and returns its result.\n *\n * For detailed usage information see the documentation for\n [scope functions](https://kotlinlang.org/docs/reference/scope-functions.html#run).\n */\n@kotlin.internal.InlineOnly\npublic inline fun <R> run(block: () -> R): R {\n contract {\n callsInPlace(block, InvocationKind.EXACTLY_ONCE)\n }\n return block()\n }\n\n/**\n * Calls the specified\n function [block] with `this` value as its receiver and returns its result.\n *\n * For detailed usage information see the\n documentation for [scope functions](https://kotlinlang.org/docs/reference/scope-functions.html#run).\n */\n@kotlin.internal.InlineOnly\npublic inline fun <T, R> T.run(block: T.() -> R): R {\n contract {\n callsInPlace(block, InvocationKind.EXACTLY_ONCE)\n }\n return block()\n }\n\n/**\n * Calls the specified\n function [block] with the given [receiver] as its receiver and returns its result.\n *\n * For detailed usage information\n see the documentation for [scope functions](https://kotlinlang.org/docs/reference/scope-functions.html#with).\n */\n@kotlin.internal.InlineOnly\npublic inline fun <T, R> with(receiver: T, block: T.() -> R): R {\n contract {\n callsInPlace(block, InvocationKind.EXACTLY_ONCE)\n }\n return receiver.block()\n }\n\n/**\n * Calls the\n specified function [block] with `this` value as its receiver and returns `this` value.\n *\n * For detailed usage\n information see the documentation for [scope functions](https://kotlinlang.org/docs/reference/scope-\n functions.html#apply).\n */\n@kotlin.internal.InlineOnly\npublic inline fun <T> T.apply(block: T.() -> Unit): T {\n contract {\n callsInPlace(block, InvocationKind.EXACTLY_ONCE)\n }\n block()\n return

```

```

this\n}\n\n/**\n * Calls the specified function [block] with `this` value as its argument and returns `this` value.\n *\n * For detailed usage information see the documentation for [scope
functions](https://kotlinalang.org/docs/reference/scope-functions.html#also).\n
*\n@kotlin.internal.InlineOnly\n@SinceKotlin("1.1")\npublic inline fun <T> T.also(block: (T) -> Unit): T {\n
contract {\n  callsInPlace(block, InvocationKind.EXACTLY_ONCE)\n } \n  block(this)\n  return
this\n}\n\n/**\n * Calls the specified function [block] with `this` value as its argument and returns its result.\n *\n * For detailed usage information see the documentation for [scope
functions](https://kotlinalang.org/docs/reference/scope-functions.html#let).\n
*\n@kotlin.internal.InlineOnly\npublic
inline fun <T, R> T.let(block: (T) -> R): R {\n  contract {\n  callsInPlace(block,
InvocationKind.EXACTLY_ONCE)\n } \n  return block(this)\n}\n\n/**\n * Returns `this` value if it satisfies the
given [predicate] or `null`, if it doesn't.\n *\n * For detailed usage information see the documentation for [scope
functions](https://kotlinalang.org/docs/reference/scope-functions.html#takeif-and-takeunless).\n
*\n@kotlin.internal.InlineOnly\n@SinceKotlin("1.1")\npublic inline fun <T> T.takeIf(predicate: (T) -> Boolean):
T? {\n  contract {\n  callsInPlace(predicate, InvocationKind.EXACTLY_ONCE)\n } \n  return if
(predicate(this)) this else null\n}\n\n/**\n * Returns `this` value if it _does not_ satisfy the given [predicate] or
`null`, if it does.\n *\n * For detailed usage information see the documentation for [scope
functions](https://kotlinalang.org/docs/reference/scope-functions.html#takeif-and-takeunless).\n
*\n@kotlin.internal.InlineOnly\n@SinceKotlin("1.1")\npublic inline fun <T> T.takeUnless(predicate: (T) ->
Boolean): T? {\n  contract {\n  callsInPlace(predicate, InvocationKind.EXACTLY_ONCE)\n } \n  return if
(!predicate(this)) this else null\n}\n\n/**\n * Executes the given function [action] specified number of [times].\n *\n * A zero-based index of current iteration is passed as a parameter to [action].\n *\n * @sample
samples.misc.ControlFlow.repeat\n
*\n@kotlin.internal.InlineOnly\npublic inline fun repeat(times: Int, action: (Int)
-> Unit) {\n  contract { callsInPlace(action) }\n  for (index in 0 until times) {\n  action(index)\n
}\n}\n", "/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of
this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*\n@kotlin.package.kotlin.comparisons\n\n/\n// NOTE: THIS FILE IS AUTO-GENERATED by the
GenerateStandardLib.kt\n// See: https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\nimport
kotlin.js.*\n\n/**\n * Returns the greater of two values.\n *\n * If values are equal, returns the first one.\n
*\n@SinceKotlin("1.1")\npublic actual fun <T : Comparable<T>> maxOf(a: T, b: T): T {\n  return if (a >= b) a
else b\n}\n\n/**\n * Returns the greater of two values.\n
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun maxOf(a: Byte, b: Byte): Byte {\n
return maxOf(a.toInt(), b.toInt()).unsafeCast<Byte>()\n}\n\n/**\n * Returns the greater of two values.\n
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun maxOf(a: Short, b: Short): Short
{\n  return maxOf(a.toInt(), b.toInt()).unsafeCast<Short>()\n}\n\n/**\n * Returns the greater of two values.\n
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun maxOf(a: Int, b: Int): Int {\n
return JsMath.max(a, b)\n}\n\n/**\n * Returns the greater of two values.\n
*\n@SinceKotlin("1.1")\n@Suppress("NOTHING_TO_INLINE")\npublic actual inline fun maxOf(a: Long, b:
Long): Long {\n  return if (a >= b) a else b\n}\n\n/**\n * Returns the greater of two values.\n *\n * If either value
is `NaN`, returns `NaN`.\n
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun
maxOf(a: Float, b: Float): Float {\n  return JsMath.max(a, b)\n}\n\n/**\n * Returns the greater of two values.\n *\n * If either value is `NaN`, returns `NaN`.\n
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual
inline fun maxOf(a: Double, b: Double): Double {\n  return JsMath.max(a, b)\n}\n\n/**\n * Returns the greater of
three values.\n *\n * If there are multiple equal maximal values, returns the first of them.\n
*\n@SinceKotlin("1.1")\npublic actual fun <T : Comparable<T>> maxOf(a: T, b: T, c: T): T {\n  return
maxOf(a, maxOf(b, c))\n}\n\n/**\n * Returns the greater of three values.\n
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun maxOf(a: Byte, b: Byte, c: Byte):
Byte {\n  return JsMath.max(a.toInt(), b.toInt(), c.toInt()).unsafeCast<Byte>()\n}\n\n/**\n * Returns the greater of
three values.\n
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun maxOf(a: Short, b:

```

Short, c: Short): Short {\n return JsMath.max(a.toInt(), b.toInt(), c.toInt()).unsafeCast<Short>()\n}\n\n/\*\*\n \* Returns the greater of three values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun maxOf(a: Int, b: Int, c: Int): Int {\n return JsMath.max(a, b, c)\n}\n\n/\*\*\n \* Returns the greater of three values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun maxOf(a: Long, b: Long, c: Long): Long {\n return maxOf(a, maxOf(b, c))\n}\n\n/\*\*\n \* Returns the greater of three values.\n \* \n \* If any value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun maxOf(a: Float, b: Float, c: Float): Float {\n return JsMath.max(a, b, c)\n}\n\n/\*\*\n \* Returns the greater of three values.\n \* \n \* If any value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun maxOf(a: Double, b: Double, c: Double): Double {\n return JsMath.max(a, b, c)\n}\n\n/\*\*\n \* Returns the greater of the given values.\n \* \n \* If there are multiple equal maximal values, returns the first of them.\n \*/\n@SinceKotlin("1.4")\npublic actual fun <T : Comparable<T>> maxOf(a: T, vararg other: T): T {\n var max = a\n for (e in other) max = maxOf(max, e)\n return max\n}\n\n/\*\*\n \* Returns the greater of the given values.\n \*/\n@SinceKotlin("1.4")\npublic actual fun maxOf(a: Byte, vararg other: Byte): Byte {\n var max = a\n for (e in other) max = maxOf(max, e)\n return max\n}\n\n/\*\*\n \* Returns the greater of the given values.\n \*/\n@SinceKotlin("1.4")\npublic actual fun maxOf(a: Short, vararg other: Short): Short {\n var max = a\n for (e in other) max = maxOf(max, e)\n return max\n}\n\n/\*\*\n \* Returns the greater of the given values.\n \*/\n@SinceKotlin("1.4")\npublic actual fun maxOf(a: Int, vararg other: Int): Int {\n var max = a\n for (e in other) max = maxOf(max, e)\n return max\n}\n\n/\*\*\n \* Returns the greater of the given values.\n \*/\n@SinceKotlin("1.4")\npublic actual fun maxOf(a: Long, vararg other: Long): Long {\n var max = a\n for (e in other) max = maxOf(max, e)\n return max\n}\n\n/\*\*\n \* Returns the greater of the given values.\n \* \n \* If any value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.4")\npublic actual fun maxOf(a: Float, vararg other: Float): Float {\n var max = a\n for (e in other) max = maxOf(max, e)\n return max\n}\n\n/\*\*\n \* Returns the greater of the given values.\n \* \n \* If any value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.4")\npublic actual fun maxOf(a: Double, vararg other: Double): Double {\n var max = a\n for (e in other) max = maxOf(max, e)\n return max\n}\n\n/\*\*\n \* Returns the smaller of two values.\n \* \n \* If values are equal, returns the first one.\n \*/\n@SinceKotlin("1.1")\npublic actual fun <T : Comparable<T>> minOf(a: T, b: T): T {\n return if (a <= b) a else b\n}\n\n/\*\*\n \* Returns the smaller of two values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun minOf(a: Byte, b: Byte): Byte {\n return minOf(a.toInt(), b.toInt()).unsafeCast<Byte>()\n}\n\n/\*\*\n \* Returns the smaller of two values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun minOf(a: Short, b: Short): Short {\n return minOf(a.toInt(), b.toInt()).unsafeCast<Short>()\n}\n\n/\*\*\n \* Returns the smaller of two values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun minOf(a: Int, b: Int): Int {\n return JsMath.min(a, b)\n}\n\n/\*\*\n \* Returns the smaller of two values.\n \*/\n@SinceKotlin("1.1")\n@Suppress("NOTHING\_TO\_INLINE")\npublic actual inline fun minOf(a: Long, b: Long): Long {\n return if (a <= b) a else b\n}\n\n/\*\*\n \* Returns the smaller of two values.\n \* \n \* If either value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun minOf(a: Float, b: Float): Float {\n return JsMath.min(a, b)\n}\n\n/\*\*\n \* Returns the smaller of two values.\n \* \n \* If either value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun minOf(a: Double, b: Double): Double {\n return JsMath.min(a, b)\n}\n\n/\*\*\n \* Returns the smaller of three values.\n \* \n \* If there are multiple equal minimal values, returns the first of them.\n \*/\n@SinceKotlin("1.1")\npublic actual fun <T : Comparable<T>> minOf(a: T, b: T, c: T): T {\n return minOf(a, minOf(b, c))\n}\n\n/\*\*\n \* Returns the smaller of three values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun minOf(a: Byte, b: Byte, c: Byte): Byte {\n return JsMath.min(a.toInt(), b.toInt(), c.toInt()).unsafeCast<Byte>()\n}\n\n/\*\*\n \* Returns the smaller of three values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun minOf(a: Short, b: Short, c: Short): Short {\n return JsMath.min(a.toInt(), b.toInt(), c.toInt()).unsafeCast<Short>()\n}\n\n/\*\*\n \* Returns the smaller of three values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun minOf(a: Int, b: Int, c: Int): Int {\n return JsMath.min(a, b, c)\n}\n\n/\*\*\n \* Returns the smaller of three

```

values.\n *\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun minOf(a: Long, b: Long,
c: Long): Long {\n    return minOf(a, minOf(b, c))\n}\n\n/**\n * Returns the smaller of three values.\n * \n * If any
value is `NaN`, returns `NaN`.\n *\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun
minOf(a: Float, b: Float, c: Float): Float {\n    return JsMath.min(a, b, c)\n}\n\n/**\n * Returns the smaller of three
values.\n * \n * If any value is `NaN`, returns `NaN`.\n
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic actual inline fun minOf(a: Double, b: Double, c:
Double): Double {\n    return JsMath.min(a, b, c)\n}\n\n/**\n * Returns the smaller of the given values.\n * \n * If
there are multiple equal minimal values, returns the first of them.\n *\n@SinceKotlin("1.4")\npublic actual fun <T
: Comparable<T>> minOf(a: T, vararg other: T): T {\n    var min = a\n    for (e in other) min = minOf(min, e)\n
return min\n}\n\n/**\n * Returns the smaller of the given values.\n *\n@SinceKotlin("1.4")\npublic actual fun
minOf(a: Byte, vararg other: Byte): Byte {\n    var min = a\n    for (e in other) min = minOf(min, e)\n    return
min\n}\n\n/**\n * Returns the smaller of the given values.\n *\n@SinceKotlin("1.4")\npublic actual fun minOf(a:
Short, vararg other: Short): Short {\n    var min = a\n    for (e in other) min = minOf(min, e)\n    return
min\n}\n\n/**\n * Returns the smaller of the given values.\n *\n@SinceKotlin("1.4")\npublic actual fun minOf(a:
Int, vararg other: Int): Int {\n    var min = a\n    for (e in other) min = minOf(min, e)\n    return min\n}\n\n/**\n
* Returns the smaller of the given values.\n *\n@SinceKotlin("1.4")\npublic actual fun minOf(a: Long, vararg
other: Long): Long {\n    var min = a\n    for (e in other) min = minOf(min, e)\n    return min\n}\n\n/**\n * Returns
the smaller of the given values.\n * \n * If any value is `NaN`, returns `NaN`.\n *\n@SinceKotlin("1.4")\npublic
actual fun minOf(a: Float, vararg other: Float): Float {\n    var min = a\n    for (e in other) min = minOf(min, e)\n
return min\n}\n\n/**\n * Returns the smaller of the given values.\n * \n * If any value is `NaN`, returns `NaN`.\n
*\n@SinceKotlin("1.4")\npublic actual fun minOf(a: Double, vararg other: Double): Double {\n    var min = a\n
for (e in other) min = minOf(min, e)\n    return min\n}\n\n"/**\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin
Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.\n *\n@n// Auto-generated file. DO NOT EDIT!\n\npackage kotlin\n\nimport
kotlin.experimental.*\nimport
kotlin.jvm.*\n\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@JvmInline\npu
blic value class ULong @PublishedApi internal constructor(@PublishedApi internal val data: Long) :
Comparable<ULong> {\n\n    companion object {\n        /**\n         * A constant holding the minimum value an
instance of ULong can have.\n         *\n         * public const val MIN_VALUE: ULong = ULong(0)\n         *\n
         * A constant holding the maximum value an instance of ULong can have.\n         *\n         * public const val
MAX_VALUE: ULong = ULong(-1)\n         *\n         * The number of bytes used to represent an instance of
ULong in a binary form.\n         *\n         * public const val SIZE_BYTES: Int = 8\n         *\n         * The number
of bits used to represent an instance of ULong in a binary form.\n         *\n         * public const val SIZE_BITS: Int = 64\n
        }\n\n        /**\n         * Compares this value with the specified value for order.\n         * Returns zero if this value is equal to
the specified other value, a negative number if it's less than other,\n         * or a positive number if it's greater than
other.\n         *\n         * @kotlin.internal.InlineOnly\n         * public inline operator fun compareTo(other: UByte): Int =
this.compareTo(other.toULong())\n         *\n         * Compares this value with the specified value for order.\n         *
Returns zero if this value is equal to the specified other value, a negative number if it's less than other,\n         * or
a positive number if it's greater than other.\n         *\n         * @kotlin.internal.InlineOnly\n         * public inline operator fun
compareTo(other: UShort): Int = this.compareTo(other.toULong())\n         *\n         * Compares this value with the
specified value for order.\n         * Returns zero if this value is equal to the specified other value, a negative number if
it's less than other,\n         * or a positive number if it's greater than other.\n         *\n         * @kotlin.internal.InlineOnly\n
         * public inline operator fun compareTo(other: UInt): Int = this.compareTo(other.toULong())\n         *\n
         * Compares this value with the specified value for order.\n         * Returns zero if this value is equal to the specified other
value, a negative number if it's less than other,\n         * or a positive number if it's greater than other.\n         *\n
         * @kotlin.internal.InlineOnly\n         * @Suppress("OVERRIDE_BY_INLINE")\n         * public override inline operator fun
compareTo(other: ULong): Int = ulongCompare(this.data, other.data)\n         *\n         * Adds the other value to this value.\n
         *\n         * @kotlin.internal.InlineOnly\n         * public inline operator fun plus(other: UByte): ULong =

```

```

this.plus(other.toULong())\n /** Adds the other value to this value. *\n @kotlin.internal.InlineOnly\n public
inline operator fun plus(other: UShort): ULong = this.plus(other.toULong())\n /** Adds the other value to this
value. *\n @kotlin.internal.InlineOnly\n public inline operator fun plus(other: UInt): ULong =
this.plus(other.toULong())\n /** Adds the other value to this value. *\n @kotlin.internal.InlineOnly\n public
inline operator fun plus(other: ULong): ULong = ULong(this.data.plus(other.data))\n\n /** Subtracts the other
value from this value. *\n @kotlin.internal.InlineOnly\n public inline operator fun minus(other: UByte): ULong
= this.minus(other.toULong())\n /** Subtracts the other value from this value. *\n @kotlin.internal.InlineOnly\n
public inline operator fun minus(other: UShort): ULong = this.minus(other.toULong())\n /** Subtracts the other
value from this value. *\n @kotlin.internal.InlineOnly\n public inline operator fun minus(other: UInt): ULong =
this.minus(other.toULong())\n /** Subtracts the other value from this value. *\n @kotlin.internal.InlineOnly\n
public inline operator fun minus(other: ULong): ULong = ULong(this.data.minus(other.data))\n\n /** Multiplies
this value by the other value. *\n @kotlin.internal.InlineOnly\n public inline operator fun times(other: UByte):
ULong = this.times(other.toULong())\n /** Multiplies this value by the other value. *\n
@kotlin.internal.InlineOnly\n public inline operator fun times(other: UShort): ULong =
this.times(other.toULong())\n /** Multiplies this value by the other value. *\n @kotlin.internal.InlineOnly\n
public inline operator fun times(other: UInt): ULong = this.times(other.toULong())\n /** Multiplies this value by
the other value. *\n @kotlin.internal.InlineOnly\n public inline operator fun times(other: ULong): ULong =
ULong(this.data.times(other.data))\n\n /** Divides this value by the other value, truncating the result to an integer
that is closer to zero. *\n @kotlin.internal.InlineOnly\n public inline operator fun div(other: UByte): ULong =
this.div(other.toULong())\n /** Divides this value by the other value, truncating the result to an integer that is
closer to zero. *\n @kotlin.internal.InlineOnly\n public inline operator fun div(other: UShort): ULong =
this.div(other.toULong())\n /** Divides this value by the other value, truncating the result to an integer that is
closer to zero. *\n @kotlin.internal.InlineOnly\n public inline operator fun div(other: UInt): ULong =
this.div(other.toULong())\n /** Divides this value by the other value, truncating the result to an integer that is
closer to zero. *\n @kotlin.internal.InlineOnly\n public inline operator fun div(other: ULong): ULong =
ulongDivide(this, other)\n\n /**\n * Calculates the remainder of truncating division of this value by the other
value.\n * \n * The result is always less than the divisor.\n *\n @kotlin.internal.InlineOnly\n public
inline operator fun rem(other: UByte): ULong = this.rem(other.toULong())\n /**\n * Calculates the remainder
of truncating division of this value by the other value.\n * \n * The result is always less than the divisor.\n
*\n @kotlin.internal.InlineOnly\n public inline operator fun rem(other: UShort): ULong =
this.rem(other.toULong())\n /**\n * Calculates the remainder of truncating division of this value by the other
value.\n * \n * The result is always less than the divisor.\n *\n @kotlin.internal.InlineOnly\n public
inline operator fun rem(other: UInt): ULong = this.rem(other.toULong())\n /**\n * Calculates the remainder of
truncating division of this value by the other value.\n * \n * The result is always less than the divisor.\n
*\n @kotlin.internal.InlineOnly\n public inline operator fun rem(other: ULong): ULong = ulongRemainder(this,
other)\n\n /**\n * Divides this value by the other value, flooring the result to an integer that is closer to negative
infinity.\n * \n * For unsigned types, the results of flooring division and truncating division are the same.\n
*\n @kotlin.internal.InlineOnly\n public inline fun floorDiv(other: UByte): ULong =
this.floorDiv(other.toULong())\n /**\n * Divides this value by the other value, flooring the result to an integer
that is closer to negative infinity.\n * \n * For unsigned types, the results of flooring division and truncating
division are the same.\n *\n @kotlin.internal.InlineOnly\n public inline fun floorDiv(other: UShort): ULong
= this.floorDiv(other.toULong())\n /**\n * Divides this value by the other value, flooring the result to an integer
that is closer to negative infinity.\n * \n * For unsigned types, the results of flooring division and truncating
division are the same.\n *\n @kotlin.internal.InlineOnly\n public inline fun floorDiv(other: UInt): ULong =
this.floorDiv(other.toULong())\n /**\n * Divides this value by the other value, flooring the result to an integer
that is closer to negative infinity.\n * \n * For unsigned types, the results of flooring division and truncating
division are the same.\n *\n @kotlin.internal.InlineOnly\n public inline fun floorDiv(other: ULong): ULong =
div(other)\n\n /**\n * Calculates the remainder of flooring division of this value by the other value.\n * \n *

```

The result is always less than the divisor.

```

    * For unsigned types, the remainders of flooring division and
    truncating division are the same.
    @kotlin.internal.InlineOnly
    public inline fun mod(other: UByte):
    UByte = this.mod(other.toULong()).toUByte()
    /**
    * Calculates the remainder of flooring division of this
    value by the other value.
    * The result is always less than the divisor.
    * For unsigned types, the remainders of flooring division and
    truncating division are the same.
    @kotlin.internal.InlineOnly
    public inline fun mod(other: UShort): UShort = this.mod(other.toULong()).toUShort()
    /**
    * Calculates the
    remainder of flooring division of this value by the other value.
    * The result is always less than the
    divisor.
    * For unsigned types, the remainders of flooring division and truncating division are the same.
    @kotlin.internal.InlineOnly
    public inline fun mod(other: UInt): UInt =
    this.mod(other.toULong()).toUInt()
    /**
    * Calculates the remainder of flooring division of this value by the
    other value.
    * The result is always less than the divisor.
    * For unsigned types, the remainders
    of flooring division and truncating division are the same.
    @kotlin.internal.InlineOnly
    public inline
    fun mod(other: ULong): ULong = rem(other)
    /**
    * Returns this value incremented by one.
    @sample samples.misc.Builtins.inc
    @kotlin.internal.InlineOnly
    public inline operator fun inc():
    ULong = ULong(data.inc())
    /**
    * Returns this value decremented by one.
    @sample
    samples.misc.Builtins.dec
    @kotlin.internal.InlineOnly
    public inline operator fun dec(): ULong =
    ULong(data.dec())
    /**
    * Creates a range from this value to the specified [other] value.
    @kotlin.internal.InlineOnly
    public inline operator fun rangeTo(other: ULong): ULongRange =
    ULongRange(this, other)
    /**
    * Shifts this value left by the [bitCount] number of bits.
    * Note
    that only the six lowest-order bits of the [bitCount] are used as the shift distance.
    * The shift distance actually
    used is therefore always in the range `0..63`.
    @kotlin.internal.InlineOnly
    public inline infix fun
    shl(bitCount: Int): ULong = ULong(data shl bitCount)
    /**
    * Shifts this value right by the [bitCount]
    number of bits, filling the leftmost bits with zeros.
    * Note that only the six lowest-order bits of the
    [bitCount] are used as the shift distance.
    * The shift distance actually used is therefore always in the range
    `0..63`.
    @kotlin.internal.InlineOnly
    public inline infix fun shr(bitCount: Int): ULong = ULong(data
    ushr bitCount)
    /**
    * Performs a bitwise AND operation between the two values.
    @kotlin.internal.InlineOnly
    public inline infix fun and(other: ULong): ULong = ULong(this.data and
    other.data)
    /**
    * Performs a bitwise OR operation between the two values.
    @kotlin.internal.InlineOnly
    public inline infix fun or(other: ULong): ULong = ULong(this.data or other.data)
    /**
    * Performs a bitwise XOR
    operation between the two values.
    @kotlin.internal.InlineOnly
    public inline infix fun xor(other: ULong):
    ULong = ULong(this.data xor other.data)
    /**
    * Inverts the bits in this value.
    @kotlin.internal.InlineOnly
    public inline fun inv(): ULong = ULong(data.inv())
    /**
    * Converts this [ULong] value to [Byte].
    * If this value is less than or equals to [Byte.MAX_VALUE], the resulting `Byte` value represents
    * the same
    numerical value as this `ULong`.
    * The resulting `Byte` value is represented by the least significant 8 bits
    of this `ULong` value.
    * Note that the resulting `Byte` value may be negative.
    @kotlin.internal.InlineOnly
    public inline fun toByte(): Byte = data.toByte()
    /**
    * Converts this [ULong]
    value to [Short].
    * If this value is less than or equals to [Short.MAX_VALUE], the resulting `Short` value
    represents
    * the same numerical value as this `ULong`.
    * The resulting `Short` value is represented
    by the least significant 16 bits of this `ULong` value.
    * Note that the resulting `Short` value may be negative.
    @kotlin.internal.InlineOnly
    public inline fun toShort(): Short = data.toShort()
    /**
    * Converts this
    [ULong] value to [Int].
    * If this value is less than or equals to [Int.MAX_VALUE], the resulting `Int`
    value represents
    * the same numerical value as this `ULong`.
    * The resulting `Int` value is
    represented by the least significant 32 bits of this `ULong` value.
    * Note that the resulting `Int` value may be
    negative.
    @kotlin.internal.InlineOnly
    public inline fun toInt(): Int = data.toInt()
    /**
    *
    Converts this [ULong] value to [Long].
    * If this value is less than or equals to [Long.MAX_VALUE], the
    resulting `Long` value represents
    * the same numerical value as this `ULong`. Otherwise the result is
    negative.
    * The resulting `Long` value has the same binary representation as this `ULong` value.
    @kotlin.internal.InlineOnly
    public inline fun toLong(): Long = data
    /**
    * Converts this [ULong]

```

```

value to [UByte].\n * If this value is less than or equals to [UByte.MAX_VALUE], the resulting `UByte`
value represents\n * the same numerical value as this `ULong`.\n * The resulting `UByte` value is
represented by the least significant 8 bits of this `ULong` value.\n * @kotlin.internal.InlineOnly\n public
inline fun toUByte(): UByte = data.toUByte()\n /**\n * Converts this [ULong] value to [UShort].\n * If
this value is less than or equals to [UShort.MAX_VALUE], the resulting `UShort` value represents\n * the
same numerical value as this `ULong`.\n * The resulting `UShort` value is represented by the least
significant 16 bits of this `ULong` value.\n * @kotlin.internal.InlineOnly\n public inline fun
toUShort(): UShort = data.toUShort()\n /**\n * Converts this [ULong] value to [UInt].\n * If this
value is less than or equals to [UInt.MAX_VALUE], the resulting `UInt` value represents\n * the same
numerical value as this `ULong`.\n * The resulting `UInt` value is represented by the least
significant 32 bits of this `ULong` value.\n * @kotlin.internal.InlineOnly\n public inline fun
toUInt(): UInt = data.toUInt()\n /** Returns this value. *\n @kotlin.internal.InlineOnly\n public
inline fun toULong(): ULong = this\n /**\n * Converts this [ULong]
value to [Float].\n * The resulting value is the closest `Float` to this `ULong` value.\n * In case
when this `ULong` value is exactly between two `Float`s,\n * the one with zero at least significant bit
of mantissa is selected.\n * @kotlin.internal.InlineOnly\n public inline fun toFloat(): Float =
this.toDouble().toFloat()\n /**\n * Converts this [ULong] value to [Double].\n * The resulting
value is the closest `Double` to this `ULong` value.\n * In case when this `ULong` value is
exactly between two `Double`s,\n * the one with zero at least significant bit of mantissa is
selected.\n * @kotlin.internal.InlineOnly\n public inline fun toDouble(): Double =
ulongToDouble(data)\n\n public override fun toString(): String = ulongToString(data)\n\n /**\n *
Converts this [Byte] value to [ULong].\n * If this value is positive, the resulting `ULong` value
represents the same numerical value as this `Byte`.\n * The least significant 8 bits of the resulting
`ULong` value are the same as the bits of this `Byte` value,\n * whereas the most significant 56
bits are filled with the sign bit of this value.\n * @SinceKotlin("1.5")\n @WasExperimental(ExperimentalUnsignedTypes::class)\n @kotlin.internal.InlineOnly\n
public inline fun Byte.toULong(): ULong = ULong(this.toLong())\n /**\n * Converts this [Short] value
to [ULong].\n * If this value is positive, the resulting `ULong` value represents the same numerical
value as this `Short`.\n * The least significant 16 bits of the resulting `ULong` value are the same
as the bits of this `Short` value,\n * whereas the most significant 48 bits are filled with the sign
bit of this value.\n * @SinceKotlin("1.5")\n @WasExperimental(ExperimentalUnsignedTypes::class)\n
@kotlin.internal.InlineOnly\n
public inline fun Short.toULong(): ULong = ULong(this.toLong())\n /**\n * Converts this [Int] value
to [ULong].\n * If this value is positive, the resulting `ULong` value represents the same numerical
value as this `Int`.\n * The least significant 32 bits of the resulting `ULong` value are the same
as the bits of this `Int` value,\n * whereas the most significant 32 bits are filled with the sign
bit of this value.\n * @SinceKotlin("1.5")\n @WasExperimental(ExperimentalUnsignedTypes::class)\n
@kotlin.internal.InlineOnly\n
public inline fun Int.toULong(): ULong = ULong(this.toLong())\n /**\n * Converts this [Long] value
to [ULong].\n * If this value is positive, the resulting `ULong` value represents the same numerical
value as this `Long`.\n * The resulting `ULong` value has the same binary representation as this
`Long` value.\n * @SinceKotlin("1.5")\n @WasExperimental(ExperimentalUnsignedTypes::class)\n
@kotlin.internal.InlineOnly\n
public inline fun Long.toULong(): ULong = ULong(this)\n\n /**\n * Converts this [Float] value to
[ULong].\n * The fractional part, if any, is rounded down towards zero.\n * Returns zero if this
`Float` value is negative or `NaN`, [ULong.MAX_VALUE] if it's bigger than `ULong.MAX_VALUE`.\n *
@SinceKotlin("1.5")\n @WasExperimental(ExperimentalUnsignedTypes::class)\n @kotlin.internal.InlineOnly\n
public inline fun Float.toULong(): ULong = doubleToULong(this.toDouble())\n /**\n * Converts this
[Double] value to [ULong].\n * The fractional part, if any, is rounded down towards zero.\n * Returns
zero if this `Double` value is negative or `NaN`, [ULong.MAX_VALUE] if it's bigger than
`ULong.MAX_VALUE`.\n * @SinceKotlin("1.5")\n @WasExperimental(ExperimentalUnsignedTypes::class)\n
@kotlin.internal.InlineOnly\n
public inline fun Double.toULong(): ULong = doubleToULong(this)\n", "/*\n * Copyright 2010-2021 JetBrains
s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n

```

```

*^@file:kotlin.jvm.JvmMultifileClass^@file:kotlin.jvm.JvmName("CollectionsKt")^package
kotlin.collections^// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt^ See:
https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib^import kotlin.random.^import
kotlin.ranges.contains^import kotlin.ranges.reversed^Returns 1st *element* from the list.^
Throws an [IndexOutOfBoundsException] if the size of this list is less than 1.^
*^@kotlin.internal.InlineOnly^public inline operator fun <T> List<T>.component1(): T {^ return
get(0)^}^Returns 2nd *element* from the list.^ Throws an [IndexOutOfBoundsException] if the
size of this list is less than 2.^
*^@kotlin.internal.InlineOnly^public inline operator fun <T>
List<T>.component2(): T {^ return get(1)^}^Returns 3rd *element* from the list.^ Throws an
[IndexOutOfBoundsException] if the size of this list is less than 3.^
*^@kotlin.internal.InlineOnly^public inline
operator fun <T> List<T>.component3(): T {^ return get(2)^}^Returns 4th *element* from the list.^
* ^ Throws an [IndexOutOfBoundsException] if the size of this list is less than 4.^
*^@kotlin.internal.InlineOnly^public inline operator fun <T> List<T>.component4(): T {^ return
get(3)^}^Returns 5th *element* from the list.^ * ^ Throws an [IndexOutOfBoundsException] if the
size of this list is less than 5.^
*^@kotlin.internal.InlineOnly^public inline operator fun <T>
List<T>.component5(): T {^ return get(4)^}^Returns `true` if [element] is found in the collection.^
*^public operator fun <@kotlin.internal.OnlyInputTypes T> Iterable<T>.contains(element: T): Boolean {^ if
(this is Collection)^ return contains(element)^ return indexOf(element) >= 0^}^Returns an
element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this
collection.^ * ^ @sample samples.collections.Collections.Elements.elementAt^
Iterable<T>.elementAt(index: Int): T {^ if (this is List)^ return get(index)^ return
elementAtOrElse(index) { throw IndexOutOfBoundsException("Collection doesn't contain element at index
$index.") }^}^Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if
the [index] is out of bounds of this list.^ * ^ @sample samples.collections.Collections.Elements.elementAt^
*^@kotlin.internal.InlineOnly^public inline fun <T> List<T>.elementAt(index: Int): T {^ return
get(index)^}^Returns an element at the given [index] or the result of calling the [defaultValue] function
if the [index] is out of bounds of this collection.^ * ^ @sample
samples.collections.Collections.Elements.elementAtOrElse^
*^public fun <T>
Iterable<T>.elementAtOrElse(index: Int, defaultValue: (Int) -> T): T {^ if (this is List)^ return
this.getOrElse(index, defaultValue)^ if (index < 0)^ return defaultValue(index)^ val iterator = iterator()^
var count = 0^ while (iterator.hasNext()) {^ val element = iterator.next()^ if (index == count++)^
return element^ }^ return defaultValue(index)^}^Returns an element at the given [index] or the
result of calling the [defaultValue] function if the [index] is out of bounds of this list.^ * ^ @sample
samples.collections.Collections.Elements.elementAtOrElse^
*^@kotlin.internal.InlineOnly^public inline fun
<T> List<T>.elementAtOrElse(index: Int, defaultValue: (Int) -> T): T {^ return if (index >= 0 && index <=
lastIndex) get(index) else defaultValue(index)^}^Returns an element at the given [index] or `null` if the
[index] is out of bounds of this collection.^ * ^ @sample
samples.collections.Collections.Elements.elementAtOrNull^
*^public fun <T>
Iterable<T>.elementAtOrNull(index: Int): T? {^ if (this is List)^ return this.getOrNull(index)^ if (index <
0)^ return null^ val iterator = iterator()^ var count = 0^ while (iterator.hasNext()) {^ val element =
iterator.next()^ if (index == count++)^ return element^ }^ return null^}^Returns an
element at the given [index] or `null` if the [index] is out of bounds of this list.^ * ^ @sample
samples.collections.Collections.Elements.elementAtOrNull^
*^@kotlin.internal.InlineOnly^public inline fun
<T> List<T>.elementAtOrNull(index: Int): T? {^ return this.getOrNull(index)^}^Returns the first
element matching the given [predicate], or `null` if no such element was found.^ * ^ @sample
samples.collections.Collections.Elements.find^
*^@kotlin.internal.InlineOnly^public inline fun <T>
Iterable<T>.find(predicate: (T) -> Boolean): T? {^ return firstOrNull(predicate)^}^Returns the last
element matching the given [predicate], or `null` if no such element was found.^ * ^ @sample

```

```

samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun <T>
Iterable<T>.findLast(predicate: (T) -> Boolean): T? {\n    return lastOrNull(predicate)\n}\n\n/**\n * Returns the last
element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n */\n@kotlin.internal.InlineOnly\npublic inline fun <T>
List<T>.findLast(predicate: (T) -> Boolean): T? {\n    return lastOrNull(predicate)\n}\n\n/**\n * Returns first
element.\n * @throws [NoSuchElementException] if the collection is empty.\n */\npublic fun <T>
Iterable<T>.first(): T {\n    when (this) {\n        is List -> return this.first()\n        else -> {\n            val iterator =
iterator()\n            if (!iterator.hasNext())\n                throw NoSuchElementException("Collection is empty.")\n            return iterator.next()\n        }\n    }\n}\n\n/**\n * Returns first element.\n * @throws [NoSuchElementException]
if the list is empty.\n */\npublic fun <T> List<T>.first(): T {\n    if (isEmpty())\n        throw
NoSuchElementException("List is empty.")\n    return this[0]\n}\n\n/**\n * Returns the first element matching the
given [predicate].\n * @throws [NoSuchElementException] if no such element is found.\n */\npublic inline fun <T>
Iterable<T>.first(predicate: (T) -> Boolean): T {\n    for (element in this) if (predicate(element)) return element\n
throw NoSuchElementException("Collection contains no element matching the predicate.")\n}\n\n/**\n * Returns
the first non-null value produced by [transform] function being applied to elements of this collection in iteration
order,\n * or throws [NoSuchElementException] if no non-null value was produced.\n * \n * @sample
samples.collections.Collections.Transformations.firstNotNullOf\n
*/\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Any>
Iterable<T>.firstNotNullOf(transform: (T) -> R?): R {\n    return firstNotNullOfOrNull(transform) ?: throw
NoSuchElementException("No element of the collection was transformed to a non-null value.")\n}\n\n/**\n *
Returns the first non-null value produced by [transform] function being applied to elements of this collection in
iteration order,\n * or `null` if no non-null value was produced.\n * \n * @sample
samples.collections.Collections.Transformations.firstNotNullOf\n
*/\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Any>
Iterable<T>.firstNotNullOfOrNull(transform: (T) -> R?): R? {\n    for (element in this) {\n        val result =
transform(element)\n        if (result != null) {\n            return result\n        }\n    }\n    return null\n}\n\n/**\n *
Returns the first element, or `null` if the collection is empty.\n */\npublic fun <T> Iterable<T>.firstOrNull(): T? {\n
when (this) {\n    is List -> {\n        if (isEmpty())\n            return null\n        else\n            return this[0]\n    }\n    else -> {\n        val iterator = iterator()\n        if (!iterator.hasNext())\n            return null\n        return iterator.next()\n    }\n}\n\n/**\n * Returns the first element, or `null` if the list is empty.\n */\npublic
fun <T> List<T>.firstOrNull(): T? {\n    return if (isEmpty()) null else this[0]\n}\n\n/**\n * Returns the first element
matching the given [predicate], or `null` if element was not found.\n */\npublic inline fun <T>
Iterable<T>.firstOrNull(predicate: (T) -> Boolean): T? {\n    for (element in this) if (predicate(element)) return
element\n    return null\n}\n\n/**\n * Returns an element at the given [index] or the result of calling the
[defaultValue] function if the [index] is out of bounds of this list.\n */\n@kotlin.internal.InlineOnly\npublic inline
fun <T> List<T>.getOrNull(index: Int, defaultValue: (Int) -> T): T {\n    return if (index >= 0 && index <=
lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or `null` if the
[index] is out of bounds of this list.\n * \n * @sample samples.collections.Collections.Elements.getOrNull\n
*/\npublic fun <T> List<T>.getOrNull(index: Int): T? {\n    return if (index >= 0 && index <= lastIndex) get(index)
else null\n}\n\n/**\n * Returns first index of [element], or -1 if the collection does not contain element.\n */\npublic
fun <@kotlin.internal.OnlyInputTypes T> Iterable<T>.indexOf(element: T): Int {\n    if (this is List) return
this.indexOf(element)\n    var index = 0\n    for (item in this) {\n        checkIndexOverflow(index)\n        if (element
== item)\n            return index\n        index++\n    }\n    return -1\n}\n\n/**\n * Returns first index of [element], or -1
if the list does not contain element.\n */\n@Suppress("EXTENSION_SHADOWED_BY_MEMBER") // false
warning, extension takes precedence in some cases\npublic fun <@kotlin.internal.OnlyInputTypes T>
List<T>.indexOf(element: T): Int {\n    return indexOf(element)\n}\n\n/**\n * Returns index of the first element
matching the given [predicate], or -1 if the collection does not contain such element.\n */\npublic inline fun <T>
Iterable<T>.indexOfFirst(predicate: (T) -> Boolean): Int {\n    var index = 0\n    for (item in this) {\n

```

```

checkIndexOverflow(index)\n    if (predicate(item))\n        return index\n    index++\n } \n return -
1\n}\n\n/**\n * Returns index of the first element matching the given [predicate], or -1 if the list does not contain
such element.\n */\npublic inline fun <T> List<T>.indexOfFirst(predicate: (T) -> Boolean): Int {\n    var index = 0\n    for (item in this) {\n        if (predicate(item))\n            return index\n        index++\n    }\n    return -1\n}\n\n/**\n * Returns index of the last element matching the given [predicate], or -1 if the collection does not contain such
element.\n */\npublic inline fun <T> Iterable<T>.indexOfLast(predicate: (T) -> Boolean): Int {\n    var lastIndex = -
1\n    var index = 0\n    for (item in this) {\n        checkIndexOverflow(index)\n        if (predicate(item))\n            lastIndex = index\n            index++\n    }\n    return lastIndex\n}\n\n/**\n * Returns index of the last element matching
the given [predicate], or -1 if the list does not contain such element.\n */\npublic inline fun <T>
List<T>.indexOfLast(predicate: (T) -> Boolean): Int {\n    val iterator = this.listIterator(size)\n    while
(iterator.hasPrevious()) {\n        if (predicate(iterator.previous())) {\n            return iterator.nextIndex()\n        }\n    }\n    return -1\n}\n\n/**\n * Returns the last element.\n * \n * @throws NoSuchElementException if the collection
is empty.\n * \n * @sample samples.collections.Collections.Elements.last\n */\npublic fun <T> Iterable<T>.last(): T
{\n    when (this) {\n        is List -> return this.last()\n        else -> {\n            val iterator = iterator()\n            if
(!iterator.hasNext())\n                throw NoSuchElementException("Collection is empty.")\n            var last =
iterator.next()\n            while (iterator.hasNext())\n                last = iterator.next()\n            return last\n        }\n    }\n}\n\n/**\n * Returns the last element.\n * \n * @throws NoSuchElementException if the list is empty.\n * \n *
@sample samples.collections.Collections.Elements.last\n */\npublic fun <T> List<T>.last(): T {\n    if (isEmpty())\n        throw NoSuchElementException("List is empty.")\n    return this[lastIndex]\n}\n\n/**\n * Returns the last
element matching the given [predicate].\n * \n * @throws NoSuchElementException if no such element is found.\n * \n *
@sample samples.collections.Collections.Elements.last\n */\npublic inline fun <T>
Iterable<T>.last(predicate: (T) -> Boolean): T {\n    var last: T? = null\n    var found = false\n    for (element in this)
{\n        if (predicate(element)) {\n            last = element\n            found = true\n        }\n    }\n    if (!found) throw
NoSuchElementException("Collection contains no element matching the predicate.")\n    @Suppress("UNCHECKED_CAST")\n    return last as T\n}\n\n/**\n * Returns the last element matching the
given [predicate].\n * \n * @throws NoSuchElementException if no such element is found.\n * \n * @sample
samples.collections.Collections.Elements.last\n */\npublic inline fun <T> List<T>.last(predicate: (T) -> Boolean): T
{\n    val iterator = this.listIterator(size)\n    while (iterator.hasPrevious()) {\n        val element = iterator.previous()\n        if
(predicate(element)) return element\n    }\n    throw NoSuchElementException("List contains no element
matching the predicate.")\n}\n\n/**\n * Returns last index of [element], or -1 if the collection does not contain
element.\n */\npublic fun <@kotlin.internal.OnlyInputTypes T> Iterable<T>.lastIndexOf(element: T): Int {\n    if
(this is List) return this.lastIndexOf(element)\n    var lastIndex = -1\n    var index = 0\n    for (item in this) {\n        checkIndexOverflow(index)\n        if (element == item)\n            lastIndex = index\n            index++\n    }\n    return
lastIndex\n}\n\n/**\n * Returns last index of [element], or -1 if the list does not contain element.\n * \n *
@Suppress("EXTENSION_SHADOWED_BY_MEMBER") // false warning, extension takes precedence in
some cases\n */\npublic fun <@kotlin.internal.OnlyInputTypes T> List<T>.lastIndexOf(element: T): Int {\n    return
lastIndexOf(element)\n}\n\n/**\n * Returns the last element, or `null` if the collection is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n */\npublic fun <T> Iterable<T>.lastOrNull(): T? {\n    when (this)
{\n        is List -> return if (isEmpty()) null else this[size - 1]\n        else -> {\n            val iterator = iterator()\n            if
(!iterator.hasNext())\n                return null\n            var last = iterator.next()\n            while (iterator.hasNext())\n                last = iterator.next()\n            return last\n        }\n    }\n}\n\n/**\n * Returns the last element, or `null` if the
list is empty.\n * \n * @sample samples.collections.Collections.Elements.last\n */\npublic fun <T>
List<T>.lastOrNull(): T? {\n    return if (isEmpty()) null else this[size - 1]\n}\n\n/**\n * Returns the last element
matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n */\npublic inline fun <T> Iterable<T>.lastOrNull(predicate: (T) ->
Boolean): T? {\n    var last: T? = null\n    for (element in this) {\n        if (predicate(element)) {\n            last =
element\n        }\n    }\n    return last\n}\n\n/**\n * Returns the last element matching the given [predicate], or `null`
if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.last\n */\npublic inline

```

```

fun <T> List<T>.lastOrNull(predicate: (T) -> Boolean): T? {
    val iterator = this.listIterator(size)
    while (iterator.hasPrevious()) {
        val element = iterator.previous()
        if (predicate(element)) return element
    }
    return null
}

/** Returns a random element from this collection. */
@throws NoSuchElementException if this collection is empty.

@SinceKotlin("1.3")
@kotlin.internal.InlineOnly
public inline fun <T> Collection<T>.random(): T {
    return random(Random)
}

/** Returns a random element from this collection using the specified source of randomness. */
@throws NoSuchElementException if this collection is empty.

@SinceKotlin("1.3")
public fun <T> Collection<T>.random(random: Random): T {
    if (isEmpty())
        throw NoSuchElementException("Collection is empty.")
    return elementAt(random.nextInt(size))
}

/** Returns a random element from this collection, or `null` if this collection is empty. */

@SinceKotlin("1.4")
@WasExperimental(ExperimentalStdlibApi::class)
@kotlin.internal.InlineOnly
public inline fun <T> Collection<T>.randomOrNull(): T? {
    return randomOrNull(Random)
}

/** Returns a random element from this collection using the specified source of randomness, or `null` if this collection is empty. */

@SinceKotlin("1.4")
@WasExperimental(ExperimentalStdlibApi::class)
public fun <T> Collection<T>.randomOrNull(random: Random): T? {
    if (isEmpty())
        return null
    return elementAt(random.nextInt(size))
}

/** Returns the single element, or throws an exception if the collection is empty or has more than one element. */

@kotlin.internal.InlineOnly
public fun <T> Iterable<T>.single(): T {
    when (this) {
        is List -> return this.single()
        else -> {
            val iterator = iterator()
            if (!iterator.hasNext())
                throw NoSuchElementException("Collection is empty.")
            val single = iterator.next()
            if (iterator.hasNext())
                throw IllegalArgumentException("Collection has more than one element.")
            return single
        }
    }
}

/** Returns the single element, or throws an exception if the list is empty or has more than one element. */

@kotlin.internal.InlineOnly
public fun <T> List<T>.single(): T {
    return when (size) {
        0 -> throw NoSuchElementException("List is empty.")
        1 -> this[0]
        else -> throw IllegalArgumentException("List has more than one element.")
    }
}

/** Returns the single element matching the given [predicate], or throws exception if there is no or more than one matching element. */

@kotlin.internal.InlineOnly
public inline fun <T> Iterable<T>.single(predicate: (T) -> Boolean): T {
    var single: T? = null
    var found = false
    for (element in this) {
        if (predicate(element)) {
            if (found)
                throw IllegalArgumentException("Collection contains more than one matching element.")
            single = element
            found = true
        }
    }
    if (!found)
        throw NoSuchElementException("Collection contains no element matching the predicate.")
    @Suppress("UNCHECKED_CAST")
    return single as T
}

/** Returns single element, or `null` if the collection is empty or has more than one element. */

@kotlin.internal.InlineOnly
public fun <T> Iterable<T>.singleOrNull(): T? {
    when (this) {
        is List -> return if (size == 1) this[0] else null
        else -> {
            val iterator = iterator()
            if (!iterator.hasNext())
                return null
            val single = iterator.next()
            if (iterator.hasNext())
                return null
            return single
        }
    }
}

/** Returns single element, or `null` if the list is empty or has more than one element. */

@kotlin.internal.InlineOnly
public fun <T> List<T>.singleOrNull(): T? {
    return if (size == 1) this[0] else null
}

/** Returns the single element matching the given [predicate], or `null` if element was not found or more than one element was found. */

@kotlin.internal.InlineOnly
public inline fun <T> Iterable<T>.singleOrNull(predicate: (T) -> Boolean): T? {
    var single: T? = null
    var found = false
    for (element in this) {
        if (predicate(element)) {
            if (found)
                return null
            single = element
            found = true
        }
    }
    if (!found)
        return null
    return single
}

/** Returns a list containing all elements except first [n] elements. */
@throws IllegalArgumentException if [n] is negative.
@sample samples.collections.Collections.Transformations.drop

@kotlin.internal.InlineOnly
public fun <T> Iterable<T>.drop(n: Int): List<T> {
    require(n >= 0) { "Requested element count $n is less than zero." }
    if (n == 0)
        return toList()
    val list: ArrayList<T>
    if (this is Collection<*>) {
        val resultSize = size - n
        if (resultSize <= 0)
            return emptyList()
        if (resultSize == 1)
            return listOf(last())
        list = ArrayList<T>(resultSize)
    }
    if (this is List<T>) {
        if (this is RandomAccess)
            for (index in n until size)
                list.add(this[index])
        else
            for (item in listIterator(n))
                list.add(item)
    }
    return list
}
}
else {
    list = ArrayList<T>()
    var count = 0
    for (item in this) {

```

```

    if (count >= n) list.add(item) else ++count\n    }\n    return list.optimizeReadOnlyList()\n}\n\n/**\n * Returns a list
containing all elements except last [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n *
\n * @sample samples.collections.Collections.Transformations.drop\n */\npublic fun <T> List<T>.dropLast(n: Int):
List<T> {\n    require(n >= 0) { \"Requested element count $n is less than zero.\" }\n    return take((size -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a list containing all elements except last elements that satisfy the given
[predicate].\n * \n * @sample samples.collections.Collections.Transformations.drop\n */\npublic inline fun <T>
List<T>.dropLastWhile(predicate: (T) -> Boolean): List<T> {\n    if (!isEmpty()) {\n        val iterator =
listIterator(size)\n        while (iterator.hasPrevious()) {\n            if (!predicate(iterator.previous())) {\n                return
take(iterator.nextIndex() + 1)\n            }\n        }\n    }\n    return emptyList()\n}\n\n/**\n * Returns a list containing
all elements except first elements that satisfy the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n */\npublic inline fun <T> Iterable<T>.dropWhile(predicate:
(T) -> Boolean): List<T> {\n    var yielding = false\n    val list = ArrayList<T>()\n    for (item in this)\n        if
(yielding)\n            list.add(item)\n        else if (!predicate(item)) {\n            list.add(item)\n            yielding = true\n        }\n    return list\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * \n *
@sample samples.collections.Collections.Filtering.filter\n */\npublic inline fun <T> Iterable<T>.filter(predicate: (T)
-> Boolean): List<T> {\n    return filterTo(ArrayList<T>(), predicate)\n}\n\n/**\n * Returns a list containing only
elements matching the given [predicate].\n * @param [predicate] function that takes the index of an element and the
element itself\n * and returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexed\n */\npublic inline fun <T>
Iterable<T>.filterIndexed(predicate: (index: Int, T) -> Boolean): List<T> {\n    return
filterIndexedTo(ArrayList<T>(), predicate)\n}\n\n/**\n * Appends all elements matching the given [predicate] to
the given [destination].\n * @param [predicate] function that takes the index of an element and the element itself\n *
and returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexedTo\n */\npublic inline fun <T, C : MutableCollection<in T>>
Iterable<T>.filterIndexedTo(destination: C, predicate: (index: Int, T) -> Boolean): C {\n    forEachIndexed { index,
element ->\n        if (predicate(index, element)) destination.add(element)\n    }\n    return destination\n}\n\n/**\n *
Returns a list containing all elements that are instances of specified type parameter R.\n * \n * @sample
samples.collections.Collections.Filtering.filterIsInstance\n */\npublic inline fun <reified R>
Iterable<*>.filterIsInstance(): List<@kotlin.internal.NoInfer R> {\n    return
filterIsInstanceTo(ArrayList<R>())\n}\n\n/**\n * Appends all elements that are instances of specified type
parameter R to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterIsInstanceTo\n */\npublic inline fun <reified R, C :
MutableCollection<in R>> Iterable<*>.filterIsInstanceTo(destination: C): C {\n    for (element in this) if (element is
R) destination.add(element)\n    return destination\n}\n\n/**\n * Returns a list containing all elements not matching
the given [predicate].\n * \n * @sample samples.collections.Collections.Filtering.filter\n */\npublic inline fun <T>
Iterable<T>.filterNot(predicate: (T) -> Boolean): List<T> {\n    return filterNotTo(ArrayList<T>(),
predicate)\n}\n\n/**\n * Returns a list containing all elements that are not `null`.\n * \n * @sample
samples.collections.Collections.Filtering.filterNotNull\n */\npublic fun <T : Any> Iterable<T?>.filterNotNull():
List<T> {\n    return filterNotNullTo(ArrayList<T>())\n}\n\n/**\n * Appends all elements that are not `null` to the
given [destination].\n * \n * @sample samples.collections.Collections.Filtering.filterNotNullTo\n */\npublic fun <C
: MutableCollection<in T>, T : Any> Iterable<T?>.filterNotNullTo(destination: C): C {\n    for (element in this) if
(element != null) destination.add(element)\n    return destination\n}\n\n/**\n * Appends all elements not matching
the given [predicate] to the given [destination].\n * \n * @sample samples.collections.Collections.Filtering.filterTo\n */\npublic inline fun <T, C : MutableCollection<in T>>
Iterable<T>.filterNotTo(destination: C, predicate: (T) ->
Boolean): C {\n    for (element in this) if (!predicate(element)) destination.add(element)\n    return
destination\n}\n\n/**\n * Appends all elements matching the given [predicate] to the given [destination].\n * \n *
@sample samples.collections.Collections.Filtering.filterTo\n */\npublic inline fun <T, C : MutableCollection<in
T>> Iterable<T>.filterTo(destination: C, predicate: (T) -> Boolean): C {\n    for (element in this) if

```

```

(predicate(element)) destination.add(element)\n return destination\n}\n\n/**\n * Returns a list containing elements
at indices in the specified [indices] range.\n */\npublic fun <T> List<T>.slice(indices: IntRange): List<T> {\n if
(indices.isEmpty()) return listOf()\n return this.subList(indices.start, indices.endInclusive + 1).toList()\n}\n\n/**\n * Returns a list containing elements at specified [indices].\n */\npublic fun <T> List<T>.slice(indices:
Iterable<Int>): List<T> {\n val size = indices.collectionSizeOrDefault(10)\n if (size == 0) return emptyList()\n
val list = ArrayList<T>(size)\n for (index in indices) {\n list.add(get(index))\n }\n return list\n}\n\n/**\n * Returns a list containing first [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n *
@sample samples.collections.Collections.Transformations.take\n */\npublic fun <T> Iterable<T>.take(n: Int):
List<T> {\n require(n >= 0) {\n "Requested element count $n is less than zero.\n }\n if (n == 0) return
emptyList()\n if (this is Collection<T>) {\n if (n >= size) return toList()\n if (n == 1) return
listOf(first())\n }\n var count = 0\n val list = ArrayList<T>(n)\n for (item in this) {\n list.add(item)\n
if (++count == n)\n break\n }\n return list.optimizeReadOnlyList()\n}\n\n/**\n * Returns a list containing
last [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.take\n */\npublic fun <T> List<T>.takeLast(n: Int): List<T> {\n
require(n >= 0) {\n "Requested element count $n is less than zero.\n }\n if (n == 0) return emptyList()\n
val size =
size\n if (n >= size) return toList()\n if (n == 1) return listOf(last())\n val list = ArrayList<T>(n)\n if (this is
RandomAccess) {\n for (index in size - n until size)\n list.add(this[index])\n } else {\n for (item in
listIterator(size - n))\n list.add(item)\n }\n return list\n}\n\n/**\n * Returns a list containing last elements
satisfying the given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n */\npublic
inline fun <T> List<T>.takeLastWhile(predicate: (T) -> Boolean): List<T> {\n if (isEmpty())\n return
emptyList()\n val iterator = listIterator(size)\n while (iterator.hasPrevious()) {\n if
(!predicate(iterator.previous())) {\n iterator.next()\n val expectedSize = size - iterator.nextIndex()\n
if (expectedSize == 0) return emptyList()\n return ArrayList<T>(expectedSize).apply {\n while
(iterator.hasNext())\n add(iterator.next())\n }\n }\n }\n return toList()\n}\n\n/**\n *
Returns a list containing first elements satisfying the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.take\n */\npublic inline fun <T> Iterable<T>.takeWhile(predicate:
(T) -> Boolean): List<T> {\n val list = ArrayList<T>()\n for (item in this) {\n if (!predicate(item))\n
break\n list.add(item)\n }\n return list\n}\n\n/**\n * Reverses elements in the list in-place.\n */\npublic
expect fun <T> MutableList<T>.reverse(): Unit\n\n/**\n * Returns a list with elements in reversed order.\n
*/\npublic fun <T> Iterable<T>.reversed(): List<T> {\n if (this is Collection && size <= 1) return toList()\n
val list = toMutableList()\n list.reverse()\n return list\n}\n\n/**\n * Randomly shuffles elements in this list in-place
using the specified [random] instance as the source of randomness.\n * \n * See:
https://en.wikipedia.org/wiki/Fisher%20%80%93Yates\_shuffle#The\_modern\_algorithm\n
*/\n\n@SinceKotlin("1.3")\npublic fun <T> MutableList<T>.shuffle(random: Random): Unit {\n for (i in lastIndex
downTo 1) {\n val j = random.nextInt(i + 1)\n this[j] = this.set(i, this[j])\n }\n}\n\n/**\n * Sorts elements
in the list in-place according to natural sort order of the value returned by specified [selector] function.\n * \n * The
sort is _stable_. It means that equal elements preserve their order relative to each other after sorting.\n */\npublic
inline fun <T, R : Comparable<R>> MutableList<T>.sortBy(crossinline selector: (T) -> R?): Unit {\n if (size > 1)
sortWith(compareBy(selector))\n}\n\n/**\n * Sorts elements in the list in-place descending according to natural sort
order of the value returned by specified [selector] function.\n * \n * The sort is _stable_. It means that equal
elements preserve their order relative to each other after sorting.\n */\npublic inline fun <T, R : Comparable<R>>
MutableList<T>.sortByDescending(crossinline selector: (T) -> R?): Unit {\n if (size > 1)
sortWith(compareByDescending(selector))\n}\n\n/**\n * Sorts elements in the list in-place descending according to
their natural sort order.\n * \n * The sort is _stable_. It means that equal elements preserve their order relative
to each other after sorting.\n */\npublic fun <T : Comparable<T>> MutableList<T>.sortDescending(): Unit {\n
sortWith(reverseOrder())\n}\n\n/**\n * Returns a list of all elements sorted according to their natural sort order.\n
* \n * The sort is _stable_. It means that equal elements preserve their order relative to each other after sorting.\n
*/\n\npublic fun <T : Comparable<T>> Iterable<T>.sorted(): List<T> {\n if (this is Collection) {\n if (size <= 1)

```

```

return this.toList()\n    @Suppress("UNCHECKED_CAST")\n    return (toTypedArray<Comparable<T>>()\n as Array<T>).apply { sort() }.asList()\n }\n return toMutableList().apply { sort() }\n}\n\n * Returns a list of\n all elements sorted according to natural sort order of the value returned by specified [selector] function.\n * \n * The\n sort is stable. It means that equal elements preserve their order relative to each other after sorting.\n * \n *\n @sample samples.collections.Collections.Sorting.sortedBy\n *\n\npublic inline fun <T, R : Comparable<R>>\n Iterable<T>.sortedBy(crossinline selector: (T) -> R?): List<T> {\n    return\n sortedWith(compareBy(selector))\n}\n\n * Returns a list of all elements sorted descending according to natural\n sort order of the value returned by specified [selector] function.\n * \n * The sort is stable. It means that equal\n elements preserve their order relative to each other after sorting.\n *\n\npublic inline fun <T, R : Comparable<R>>\n Iterable<T>.sortedByDescending(crossinline selector: (T) -> R?): List<T> {\n    return\n sortedWith(compareByDescending(selector))\n}\n\n * Returns a list of all elements sorted descending\n according to their natural sort order.\n * \n * The sort is stable. It means that equal elements preserve their order\n relative to each other after sorting.\n *\n\npublic fun <T : Comparable<T>> Iterable<T>.sortedDescending(): List<T>\n {\n    return sortedWith(reverseOrder())\n}\n\n * Returns a list of all elements sorted according to the specified\n [comparator].\n * \n * The sort is stable. It means that equal elements preserve their order relative to each other\n after sorting.\n *\n\npublic fun <T> Iterable<T>.sortedWith(comparator: Comparator<in T>): List<T> {\n    if (this is\n Collection) {\n        if (size <= 1) return this.toList()\n        @Suppress("UNCHECKED_CAST")\n        return\n (toTypedArray<Any?>() as Array<T>).apply { sortWith(comparator) }.asList()\n    }\n    return\n toMutableList().apply { sortWith(comparator) }\n}\n\n * Returns an array of Boolean containing all of the\n elements of this collection.\n *\n\npublic fun Collection<Boolean>.toBooleanArray(): BooleanArray {\n    val result\n = BooleanArray(size)\n    var index = 0\n    for (element in this)\n        result[index++] = element\n    return\n result\n}\n\n * Returns an array of Byte containing all of the elements of this collection.\n *\n\npublic fun\n Collection<Byte>.toByteArray(): ByteArray {\n    val result = ByteArray(size)\n    var index = 0\n    for (element in\n this)\n        result[index++] = element\n    return result\n}\n\n * Returns an array of Char containing all of the\n elements of this collection.\n *\n\npublic fun Collection<Char>.toCharArray(): CharArray {\n    val result =\n CharArray(size)\n    var index = 0\n    for (element in this)\n        result[index++] = element\n    return\n result\n}\n\n * Returns an array of Double containing all of the elements of this collection.\n *\n\npublic fun\n Collection<Double>.toDoubleArray(): DoubleArray {\n    val result = DoubleArray(size)\n    var index = 0\n    for\n (element in this)\n        result[index++] = element\n    return result\n}\n\n * Returns an array of Float containing\n all of the elements of this collection.\n *\n\npublic fun Collection<Float>.toFloatArray(): FloatArray {\n    val result\n = FloatArray(size)\n    var index = 0\n    for (element in this)\n        result[index++] = element\n    return\n result\n}\n\n * Returns an array of Int containing all of the elements of this collection.\n *\n\npublic fun\n Collection<Int>.toIntArray(): IntArray {\n    val result = IntArray(size)\n    var index = 0\n    for (element in this)\n        result[index++] = element\n    return result\n}\n\n * Returns an array of Long containing all of the elements\n of this collection.\n *\n\npublic fun Collection<Long>.toLongArray(): LongArray {\n    val result =\n LongArray(size)\n    var index = 0\n    for (element in this)\n        result[index++] = element\n    return\n result\n}\n\n * Returns an array of Short containing all of the elements of this collection.\n *\n\npublic fun\n Collection<Short>.toShortArray(): ShortArray {\n    val result = ShortArray(size)\n    var index = 0\n    for (element\n in this)\n        result[index++] = element\n    return result\n}\n\n * Returns a [Map] containing key-value pairs\n provided by [transform] function\n * applied to elements of the given collection.\n * \n * If any of two pairs would\n have the same key the last one gets added to the map.\n * \n * The returned map preserves the entry iteration order\n of the original collection.\n *\n *\n @sample samples.collections.Collections.Transformations.associate\n *\n\npublic\n inline fun <T, K, V> Iterable<T>.associate(transform: (T) -> Pair<K, V>): Map<K, V> {\n    val capacity =\n mapCapacity(collectionSizeOrDefault(10)).coerceAtLeast(16)\n    return associateTo(LinkedHashMap<K,\n V>(capacity), transform)\n}\n\n * Returns a [Map] containing the elements from the given collection indexed\n by the key\n * returned from [keySelector] function applied to each element.\n * \n * If any two elements would\n have the same key returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves\n the entry iteration order of the original collection.\n *\n *\n @sample

```

```

samples.collections.Collections.Transformations.associateBy\n *\npublic inline fun <T, K>
Iterable<T>.associateBy(keySelector: (T) -> K): Map<K, T> {\n    val capacity =
mapCapacity(collectionSizeOrDefault(10)).coerceAtLeast(16)\n    return associateByTo(LinkedHashMap<K,
T>(capacity), keySelector)\n}\n\n/**\n * Returns a [Map] containing the values provided by [valueTransform] and
indexed by [keySelector] functions applied to elements of the given collection.\n * \n * If any two elements would
have the same key returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves
the entry iteration order of the original collection.\n * \n * @sample
samples.collections.Collections.Transformations.associateByWithValueTransform\n *\npublic inline fun <T, K, V>
Iterable<T>.associateBy(keySelector: (T) -> K, valueTransform: (T) -> V): Map<K, V> {\n    val capacity =
mapCapacity(collectionSizeOrDefault(10)).coerceAtLeast(16)\n    return associateByTo(LinkedHashMap<K,
V>(capacity), keySelector, valueTransform)\n}\n\n/**\n * Populates and returns the [destination] mutable map with
key-value pairs,\n * where key is provided by the [keySelector] function applied to each element of the given
collection\n * and value is the element itself.\n * \n * If any two elements would have the same key returned by
[keySelector] the last one gets added to the map.\n * \n * @sample
samples.collections.Collections.Transformations.associateByTo\n *\npublic inline fun <T, K, M : MutableMap<in
K, in T>> Iterable<T>.associateByTo(destination: M, keySelector: (T) -> K): M {\n    for (element in this) {\n
destination.put(keySelector(element), element)\n    }\n    return destination\n}\n\n/**\n * Populates and returns the
[destination] mutable map with key-value pairs,\n * where key is provided by the [keySelector] function and\n *
and value is provided by the [valueTransform] function applied to elements of the given collection.\n * \n * If any two
elements would have the same key returned by [keySelector] the last one gets added to the map.\n * \n * @sample
samples.collections.Collections.Transformations.associateByToWithValueTransform\n *\npublic inline fun <T, K,
V, M : MutableMap<in K, in V>> Iterable<T>.associateByTo(destination: M, keySelector: (T) -> K,
valueTransform: (T) -> V): M {\n    for (element in this) {\n        destination.put(keySelector(element),
valueTransform(element))\n    }\n    return destination\n}\n\n/**\n * Populates and returns the [destination] mutable
map with key-value pairs\n * provided by [transform] function applied to each element of the given collection.\n *
\n * If any of two pairs would have the same key the last one gets added to the map.\n * \n * @sample
samples.collections.Collections.Transformations.associateTo\n *\npublic inline fun <T, K, V, M : MutableMap<in
K, in V>> Iterable<T>.associateTo(destination: M, transform: (T) -> Pair<K, V>): M {\n    for (element in this) {\n
destination += transform(element)\n    }\n    return destination\n}\n\n/**\n * Returns a [Map] where keys are
elements from the given collection and values are\n * produced by the [valueSelector] function applied to each
element.\n * \n * If any two elements are equal, the last one gets added to the map.\n * \n * The returned map
preserves the entry iteration order of the original collection.\n * \n * @sample
samples.collections.Collections.Transformations.associateWith\n *\n@SinceKotlin("1.3")\npublic inline fun <K,
V> Iterable<K>.associateWith(valueSelector: (K) -> V): Map<K, V> {\n    val result = LinkedHashMap<K,
V>(mapCapacity(collectionSizeOrDefault(10)).coerceAtLeast(16))\n    return associateWithTo(result,
valueSelector)\n}\n\n/**\n * Populates and returns the [destination] mutable map with key-value pairs for each
element of the given collection,\n * where key is the element itself and value is provided by the [valueSelector]
function applied to that key.\n * \n * If any two elements are equal, the last one overwrites the former value in the
map.\n * \n * @sample
samples.collections.Collections.Transformations.associateWithTo\n
*\n@SinceKotlin("1.3")\npublic inline fun <K, V, M : MutableMap<in K, in V>>
Iterable<K>.associateWithTo(destination: M, valueSelector: (K) -> V): M {\n    for (element in this) {\n
destination.put(element, valueSelector(element))\n    }\n    return destination\n}\n\n/**\n * Appends all elements to
the given [destination] collection.\n *\npublic fun <T, C : MutableCollection<in T>>
Iterable<T>.toCollection(destination: C): C {\n    for (item in this) {\n        destination.add(item)\n    }\n    return
destination\n}\n\n/**\n * Returns a new [HashSet] of all elements.\n *\npublic fun <T> Iterable<T>.toHashSet():
HashSet<T> {\n    return toCollection(HashSet<T>(mapCapacity(collectionSizeOrDefault(12))))\n}\n\n/**\n *
Returns a [List] containing all elements.\n *\npublic fun <T> Iterable<T>.toList(): List<T> {\n    if (this is
Collection) {\n        return when (size) {\n            0 -> emptyList()\n            1 -> listOf(if (this is List) get(0) else

```

```

iterator().next())\n        else -> this.toMutableList()\n    }\n }\n return
this.toMutableList().optimizeReadOnlyList()\n}\n\n/**\n * Returns a new [MutableList] filled with all elements of
this collection.\n */\npublic fun <T> Iterable<T>.toMutableList(): MutableList<T> {\n    if (this is Collection<T>)\n        return this.toMutableList()\n    return toCollection(ArrayList<T>())\n}\n\n/**\n * Returns a new [MutableList]
filled with all elements of this collection.\n */\npublic fun <T> Collection<T>.toMutableList(): MutableList<T> {\n
    return ArrayList(this)\n}\n\n/**\n * Returns a [Set] of all elements.\n */\n\n * The returned set preserves the element
iteration order of the original collection.\n */\npublic fun <T> Iterable<T>.toSet(): Set<T> {\n    if (this is
Collection) {\n        return when (size) {\n            0 -> emptySet()\n            1 -> setOf(if (this is List) this[0] else
iterator().next())\n            else -> toCollection(LinkedHashSet<T>(mapCapacity(size)))\n        }\n    }\n    return
toCollection(LinkedHashSet<T>()).optimizeReadOnlySet()\n}\n\n/**\n * Returns a single list of all elements
yielded from results of [transform] function being invoked on each element of original collection.\n */\n\n * @sample
samples.collections.Collections.Transformations.flatMap\n */\npublic inline fun <T, R>
Iterable<T>.flatMap(transform: (T) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(),
transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being
invoked on each element of original collection.\n */\n\n * @sample
samples.collections.Collections.Transformations.flatMap\n
*/\n\n @SinceKotlin("1.4")\n @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n @OverloadResolution
ByLambdaReturnType\n @kotlin.jvm.JvmName("flatMapSequence")\n public inline fun <T, R>
Iterable<T>.flatMap(transform: (T) -> Sequence<R>): List<R> {\n    return flatMapTo(ArrayList<R>(),
transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being
invoked on each element\n */\n\n * and its index in the original collection.\n */\n\n * @sample
samples.collections.Collections.Transformations.flatMapIndexed\n
*/\n\n @SinceKotlin("1.4")\n @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n @OverloadResolution
ByLambdaReturnType\n @kotlin.jvm.JvmName("flatMapIndexedIterable")\n @kotlin.internal.InlineOnly\n public
inline fun <T, R> Iterable<T>.flatMapIndexed(transform: (index: Int, T) -> Iterable<R>): List<R> {\n    return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from
results of [transform] function being invoked on each element\n */\n\n * and its index in the original collection.\n */\n\n *
@sample samples.collections.Collections.Transformations.flatMapIndexed\n
*/\n\n @SinceKotlin("1.4")\n @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n @OverloadResolution
ByLambdaReturnType\n @kotlin.jvm.JvmName("flatMapIndexedSequence")\n @kotlin.internal.InlineOnly\n publi
c inline fun <T, R> Iterable<T>.flatMapIndexed(transform: (index: Int, T) -> Sequence<R>): List<R> {\n    return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Appends all elements yielded from results of
[transform] function being invoked on each element\n */\n\n * and its index in the original collection, to the given
[destination].\n */\n\n
*/\n\n @SinceKotlin("1.4")\n @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n @OverloadResolution
ByLambdaReturnType\n @kotlin.jvm.JvmName("flatMapIndexedIterableTo")\n @kotlin.internal.InlineOnly\n publi
c inline fun <T, R, C : MutableCollection<in R>> Iterable<T>.flatMapIndexedTo(destination: C, transform: (index:
Int, T) -> Iterable<R>): C {\n    var index = 0\n    for (element in this) {\n        val list =
transform(checkIndexOverflow(index++), element)\n        destination.addAll(list)\n    }\n    return
destination\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each
element\n */\n\n * and its index in the original collection, to the given [destination].\n */\n\n
*/\n\n @SinceKotlin("1.4")\n @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n @OverloadResolution
ByLambdaReturnType\n @kotlin.jvm.JvmName("flatMapIndexedSequenceTo")\n @kotlin.internal.InlineOnly\n publi
c inline fun <T, R, C : MutableCollection<in R>> Iterable<T>.flatMapIndexedTo(destination: C, transform:
(index: Int, T) -> Sequence<R>): C {\n    var index = 0\n    for (element in this) {\n        val list =
transform(checkIndexOverflow(index++), element)\n        destination.addAll(list)\n    }\n    return
destination\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each
element of original collection, to the given [destination].\n */\n\n */\n\n public inline fun <T, R, C : MutableCollection<in

```

```

R>> Iterable<T>.flatMapTo(destination: C, transform: (T) -> Iterable<R>): C {
    for (element in this) {
        val list = transform(element)
        destination.addAll(list)
    }
    return destination
}

```

Appends all elements yielded from results of [transform] function being invoked on each element of original collection, to the given [destination].

```

@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolutionByLambdaReturnType
@kotlin.jvm.JvmName("flatMapSequenceTo")
public inline fun <T, R, C : MutableCollection<in R>> Iterable<T>.flatMapTo(destination: C, transform: (T) -> Sequence<R>): C {
    for (element in this) {
        val list = transform(element)
        destination.addAll(list)
    }
    return destination
}

```

Groups elements of the original collection by the key returned by the given [keySelector] function applied to each element and returns a map where each group key is associated with a list of corresponding elements. The returned map preserves the entry iteration order of the keys produced from the original collection.

```

@sample samples.collections.Collections.Transformations.groupBy
public inline fun <T, K> Iterable<T>.groupBy(keySelector: (T) -> K): Map<K, List<T>> {
    return groupByTo(LinkedHashMap<K, MutableList<T>>(), keySelector)
}

```

Groups values returned by the [valueTransform] function applied to each element of the original collection by the key returned by the given [keySelector] function applied to the element and returns a map where each group key is associated with a list of corresponding values. The returned map preserves the entry iteration order of the keys produced from the original collection.

```

@sample samples.collections.Collections.Transformations.groupByKeysAndValues
public inline fun <T, K, V> Iterable<T>.groupBy(keySelector: (T) -> K, valueTransform: (T) -> V): Map<K, List<V>> {
    return groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)
}

```

Groups elements of the original collection by the key returned by the given [keySelector] function applied to each element and puts to the [destination] map each group key associated with a list of corresponding elements. The returned map preserves the entry iteration order of the keys produced from the original collection.

```

@return The [destination] map.
@sample samples.collections.Collections.Transformations.groupBy
public inline fun <T, K, M : MutableMap<in K, MutableList<T>>> Iterable<T>.groupByTo(destination: M, keySelector: (T) -> K): M {
    for (element in this) {
        val key = keySelector(element)
        val list = destination.getOrPut(key) { ArrayList<T>() }
        list.add(element)
    }
    return destination
}

```

Groups values returned by the [valueTransform] function applied to each element of the original collection by the key returned by the given [keySelector] function applied to the element and puts to the [destination] map each group key associated with a list of corresponding values. The returned map preserves the entry iteration order of the keys produced from the original collection.

```

@return The [destination] map.
@sample samples.collections.Collections.Transformations.groupByKeysAndValues
public inline fun <T, K, V, M : MutableMap<in K, MutableList<V>>> Iterable<T>.groupByTo(destination: M, keySelector: (T) -> K, valueTransform: (T) -> V): M {
    for (element in this) {
        val key = keySelector(element)
        val list = destination.getOrPut(key) { ArrayList<V>() }
        list.add(valueTransform(element))
    }
    return destination
}

```

Creates a [Grouping] source from a collection to be used later with one of group-and-fold operations using the specified [keySelector] function to extract a key from each element.

```

@sample samples.collections.Grouping.groupingByEachCount
@SinceKotlin("1.1")
public inline fun <T, K> Iterable<T>.groupingBy(crossinline keySelector: (T) -> K): Grouping<T, K> {
    return object : Grouping<T, K> {
        override fun sourceIterator(): Iterator<T> = this@groupingBy.iterator()
        override fun keyOf(element: T): K = keySelector(element)
    }
}

```

Returns a list containing the results of applying the given [transform] function to each element in the original collection.

```

@sample samples.collections.Collections.Transformations.map
public inline fun <T, R> Iterable<T>.map(transform: (T) -> R): List<R> {
    return mapTo(ArrayList<R>(collectionSizeOrDefault(10)), transform)
}

```

Returns a list containing the results of applying the given [transform] function to each element and its index in the original collection.

```

@param [transform] function that takes the index of an element and the element itself
and returns the result of the transform applied to the element.
public inline fun <T, R> Iterable<T>.mapIndexed(transform: (index: Int, T) -> R): List<R> {
    return mapIndexedTo(ArrayList<R>(collectionSizeOrDefault(10)), transform)
}

```

Returns a list containing only

the non-null results of applying the given [transform] function to each element and its index in the original collection.

`mapIndexedNotNull(transform: (index: Int, T) -> R?): List<R>` {  
 return  
 mapIndexedNotNullTo(ArrayList<R>(), transform)  
 \* Applies the given [transform] function to each element and its index in the original collection and appends only the non-null results to the given [destination].

`mapIndexedNotNullTo(destination: C, transform: (index: Int, T) -> R?): C` {  
 forEachIndexed {  
 index, element -> transform(index, element)?.let { destination.add(it) }  
 }  
 return destination  
 \* Applies the given [transform] function to each element and its index in the original collection and appends the results to the given [destination].

`mapIndexedTo(destination: C, transform: (index: Int, T) -> R): C` {  
 var  
 index = 0  
 for (item in this)  
 destination.add(transform(checkIndexOverflow(index++), item))  
 return  
 destination  
 \* Returns a list containing only the non-null results of applying the given [transform] function to each element in the original collection.

`mapNotNull(transform: (T) -> R?): List<R>` {  
 return mapNotNullTo(ArrayList<R>(),  
 transform)  
 \* Applies the given [transform] function to each element in the original collection and appends only the non-null results to the given [destination].

`mapNotNullTo(destination: C, transform: (T) -> R?): C` {  
 forEach {  
 element -> transform(element)?.let { destination.add(it) }  
 }  
 return destination  
 \* Applies the given [transform] function to each element of the original collection and appends the results to the given [destination].

`mapTo(destination: C, transform: (T) -> R): C` {  
 for (item in this)  
 destination.add(transform(item))  
 return  
 destination  
 \* Returns a lazy [Iterable] that wraps each element of the original collection into an [IndexedValue] containing the index of that element and the element itself.

`withIndex(): Iterable<IndexedValue<T>>` {  
 return IndexingIterable { iterator() }  
 \* Returns a list containing only distinct elements from the given collection. Among equal elements of the given collection, only the first one will be present in the resulting list. The elements in the resulting list are in the same order as they were in the source collection.

`distinctAndDistinctBy(selector: (T) -> K): List<T>` {  
 return this.toMutableSet().toList()  
 \* Returns a list containing only elements from the given collection having distinct keys returned by the given [selector] function. Among elements of the given collection with equal keys, only the first one will be present in the resulting list. The elements in the resulting list are in the same order as they were in the source collection.

`distinctBy(selector: (T) -> K): List<T>` {  
 val set = HashSet<K>()  
 val list = ArrayList<T>()  
 for (e in this) {  
 val key = selector(e)  
 if (set.add(key))  
 list.add(e)  
 }  
 return list  
 \* Returns a set containing all elements that are contained by both this collection and the specified collection. The returned set preserves the element iteration order of the original collection. To get a set containing all elements that are contained at least in one of these collections use [union].

`intersect(other: Iterable<T>): Set<T>` {  
 val set = this.toMutableSet()  
 set.retainAll(other)  
 return set  
 \* Returns a set containing all elements that are contained by this collection and not contained by the specified collection. The returned set preserves the element iteration order of the original collection.

`subtract(other: Iterable<T>): Set<T>` {  
 val set =  
 this.toMutableSet()  
 set.removeAll(other)  
 return set  
 \* Returns a new [MutableSet] containing all

```

distinct elements from the given collection.
 * The returned set preserves the element iteration order of the
original collection.
 * public fun <T> Iterable<T>.toMutableSet(): MutableSet<T> {
    return when (this) {
        is Collection<T> -> LinkedHashSet(this)
        else -> toCollection(LinkedHashSet<T>())
    }
}

 * Returns a set containing all distinct elements from both collections.
 * The returned set preserves the element
iteration order of the original collection.
 * Those elements of the [other] collection that are unique are iterated in
the end
 * in the order of the [other] collection.
 * To get a set containing all elements that are contained in
both collections use [intersect].
 * public infix fun <T> Iterable<T>.union(other: Iterable<T>): Set<T> {
    val set = this.toMutableSet()
    set.addAll(other)
    return set
}

 * Returns `true` if all elements match the
given [predicate].
 * @sample samples.collections.Collections.Aggregates.all
 * public inline fun <T>
Iterable<T>.all(predicate: (T) -> Boolean): Boolean {
    if (this is Collection && isEmpty()) return true
    for (element in this) if (!predicate(element)) return false
    return true
}

 * Returns `true` if collection has at
least one element.
 * @sample samples.collections.Collections.Aggregates.any
 * public fun <T>
Iterable<T>.any(): Boolean {
    if (this is Collection) return !isEmpty()
    return iterator().hasNext()
}

 * Returns `true` if at least one element matches the given [predicate].
 * @sample
samples.collections.Collections.Aggregates.anyWithPredicate
 * public inline fun <T>
Iterable<T>.any(predicate: (T) -> Boolean): Boolean {
    if (this is Collection && isEmpty()) return false
    for (element in this) if (predicate(element)) return true
    return false
}

 * Returns the number of elements in
this collection.
 * public fun <T> Iterable<T>.count(): Int {
    if (this is Collection) return size
    var count = 0
    for (element in this) checkCountOverflow(++count)
    return count
}

 * Returns the number of
elements in this collection.
 * @kotlin.internal.InlineOnly
 * public inline fun <T> Collection<T>.count(): Int {
    return size
}

 * Returns the number of elements matching the given [predicate].
 * public inline fun <T>
Iterable<T>.count(predicate: (T) -> Boolean): Int {
    if (this is Collection && isEmpty()) return 0
    var count = 0
    for (element in this) if (predicate(element)) checkCountOverflow(++count)
    return count
}

 * Accumulates value starting with [initial] value and applying [operation] from left to right
 * to current accumulator
value and each element.
 * Returns the specified [initial] value if the collection is empty.
 * @param
[operation] function that takes current accumulator value and an element, and calculates the next accumulator
value.
 * public inline fun <T, R> Iterable<T>.fold(initial: R, operation: (acc: R, T) -> R): R {
    var accumulator = initial
    for (element in this) accumulator = operation(accumulator, element)
    return accumulator
}

 * Accumulates value starting with [initial] value and applying [operation] from left to
right
 * to current accumulator value and each element with its index in the original collection.
 * Returns the
specified [initial] value if the collection is empty.
 * @param [operation] function that takes the index of an
element, current accumulator value
 * and the element itself, and calculates the next accumulator value.
 * public inline fun <T, R> Iterable<T>.foldIndexed(initial: R, operation: (index: Int, acc: R, T) -> R): R {
    var index = 0
    var accumulator = initial
    for (element in this) accumulator =
operation(checkIndexOverflow(index++), accumulator, element)
    return accumulator
}

 * Accumulates
value starting with [initial] value and applying [operation] from right to left
 * to each element and current
accumulator value.
 * Returns the specified [initial] value if the list is empty.
 * @param [operation]
function that takes an element and current accumulator value, and calculates the next accumulator value.
 * public inline fun <T, R> List<T>.foldRight(initial: R, operation: (T, acc: R) -> R): R {
    var accumulator =
initial
    if (!isEmpty()) {
        val iterator = listIterator(size)
        while (iterator.hasPrevious()) {
            accumulator = operation(iterator.previous(), accumulator)
        }
    }
    return accumulator
}

 * Accumulates value starting with [initial] value and applying [operation] from right to left
 * to each element with
its index in the original list and current accumulator value.
 * Returns the specified [initial] value if the list is
empty.
 * @param [operation] function that takes the index of an element, the element itself
 * and current
accumulator value, and calculates the next accumulator value.
 * public inline fun <T, R>
List<T>.foldRightIndexed(initial: R, operation: (index: Int, T, acc: R) -> R): R {
    var accumulator = initial
    if (!isEmpty()) {
        val iterator = listIterator(size)
        while (iterator.hasPrevious()) {
            val index =
iterator.previousIndex()
            accumulator = operation(index, iterator.previous(), accumulator)
        }
    }
}

```

```

return accumulator\n}\n\n/**\n * Performs the given [action] on each element.\n
*\n\n@kotlin.internal.HidesMembers\npublic inline fun <T> Iterable<T>.forEach(action: (T) -> Unit): Unit {\n for
(element in this) action(element)\n}\n\n/**\n * Performs the given [action] on each element, providing sequential
index with the element.\n * @param [action] function that takes the index of an element and the element itself\n *
and performs the action on the element.\n *\n\npublic inline fun <T> Iterable<T>.forEachIndexed(action: (index: Int,
T) -> Unit): Unit {\n var index = 0\n for (item in this) action(checkIndexOverflow(index++),
item)\n}\n\n@Deprecated("Use maxOrNull instead.")
ReplaceWith("this.maxOrNull()")\n\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\n\n@SinceKotlin("1.1")\npublic fun Iterable<Double>.max(): Double? {\n return
maxOrNull()\n}\n\n@Deprecated("Use maxOrNull instead.")
ReplaceWith("this.maxOrNull()")\n\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\n\n@SinceKotlin("1.1")\npublic fun Iterable<Float>.max(): Float? {\n return
maxOrNull()\n}\n\n@Deprecated("Use maxOrNull instead.")
ReplaceWith("this.maxOrNull()")\n\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\n\npublic fun <T : Comparable<T>> Iterable<T>.max(): T? {\n return
maxOrNull()\n}\n\n@Deprecated("Use maxByOrNull instead.")
ReplaceWith("this.maxByOrNull(selector)")\n\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\n\npublic inline fun <T, R : Comparable<R>> Iterable<T>.maxBy(selector: (T) -> R):
T? {\n return maxByOrNull(selector)\n}\n\n/**\n * Returns the first element yielding the largest value of the given
function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.maxByOrNull\n *\n\n@SinceKotlin("1.4")\n\npublic inline fun <T, R :
Comparable<R>> Iterable<T>.maxByOrNull(selector: (T) -> R): T? {\n val iterator = iterator()\n if
(!iterator.hasNext()) return null\n var maxElem = iterator.next()\n if (!iterator.hasNext()) return maxElem\n var
maxValue = selector(maxElem)\n do {\n val e = iterator.next()\n val v = selector(e)\n if (maxValue <
v) {\n maxElem = e\n maxValue = v\n }\n } while (iterator.hasNext())\n return
maxElem\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each element in the collection.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result
is `NaN`.\n * \n * @throws NoSuchElementException if the collection is empty.\n
*\n\n@SinceKotlin("1.4")\n\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n\n@OverloadResolution
ByLambdaReturnType\n\n@kotlin.internal.InlineOnly\n\npublic inline fun <T> Iterable<T>.maxOf(selector: (T) ->
Double): Double {\n val iterator = iterator()\n if (!iterator.hasNext()) throw NoSuchElementException()\n var
maxValue = selector(iterator.next())\n while (iterator.hasNext()) {\n val v = selector(iterator.next())\n
maxValue = maxOf(maxValue, v)\n }\n return maxValue\n}\n\n/**\n * Returns the largest value among all
values produced by [selector] function\n * applied to each element in the collection.\n * \n * If any of values
produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException
if the collection is empty.\n
*\n\n@SinceKotlin("1.4")\n\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n\n@OverloadResolution
ByLambdaReturnType\n\n@kotlin.internal.InlineOnly\n\npublic inline fun <T> Iterable<T>.maxOf(selector: (T) ->
Float): Float {\n val iterator = iterator()\n if (!iterator.hasNext()) throw NoSuchElementException()\n var
maxValue = selector(iterator.next())\n while (iterator.hasNext()) {\n val v = selector(iterator.next())\n
maxValue = maxOf(maxValue, v)\n }\n return maxValue\n}\n\n/**\n * Returns the largest value among all
values produced by [selector] function\n * applied to each element in the collection.\n * \n * @throws
NoSuchElementException if the collection is empty.\n
*\n\n@SinceKotlin("1.4")\n\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n\n@OverloadResolution
ByLambdaReturnType\n\n@kotlin.internal.InlineOnly\n\npublic inline fun <T, R : Comparable<R>>
Iterable<T>.maxOf(selector: (T) -> R): R {\n val iterator = iterator()\n if (!iterator.hasNext()) throw
NoSuchElementException()\n var maxValue = selector(iterator.next())\n while (iterator.hasNext()) {\n val v
= selector(iterator.next())\n if (maxValue < v) {\n maxValue = v\n }\n }\n return

```

`maxValue` Returns the largest value among all values produced by [selector] function applied to each element in the collection or `null` if there are no elements. If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Iterable<T>.maxOrNull(selector: (T)\n-> Double): Double? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var maxValue =\n    selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v = selector(iterator.next())\n        maxValue =\n        maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n* Returns the largest value among all values produced\nby [selector] function applied to each element in the collection or `null` if there are no elements.\n* If any\nof values produced by [selector] function is `NaN`, the returned result is `NaN`.
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Iterable<T>.maxOrNull(selector: (T)\n-> Float): Float? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var maxValue =\n    selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v = selector(iterator.next())\n        maxValue =\n        maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n* Returns the largest value among all values produced\nby [selector] function applied to each element in the collection or `null` if there are no elements.
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Comparable<R>>\nIterable<T>.maxOrNull(selector: (T) -> R): R? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return\n    null\n    var maxValue = selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v =\n        selector(iterator.next())\n        if (maxValue < v) {\n            maxValue = v\n        }\n    }\n    return\n    maxValue\n}\n\n* Returns the largest value according to the provided [comparator]\n* among all values\nproduced by [selector] function applied to each element in the collection.\n* @throws\nNoSuchElementException if the collection is empty.
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R>\nIterable<T>.maxWith(comparator: Comparator<in R>, selector: (T) -> R): R {\n    val iterator = iterator()\n    if\n    (!iterator.hasNext()) throw NoSuchElementException()\n    var maxValue = selector(iterator.next())\n    while\n    (iterator.hasNext()) {\n        val v = selector(iterator.next())\n        if (comparator.compare(maxValue, v) < 0) {\n           \n            maxValue = v\n        }\n    }\n    return maxValue\n}\n\n* Returns the largest value according to the provided\n[comparator]\n* among all values produced by [selector] function applied to each element in the collection or `null`\nif there are no elements.
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R>\nIterable<T>.maxWithOrNull(comparator: Comparator<in R>, selector: (T) -> R): R? {\n    val iterator =\n    iterator()\n    if (!iterator.hasNext()) return null\n    var maxValue = selector(iterator.next())\n    while\n    (iterator.hasNext()) {\n        val v = selector(iterator.next())\n        if (comparator.compare(maxValue, v) < 0) {\n           \n            maxValue = v\n        }\n    }\n    return maxValue\n}\n\n* Returns the largest element or `null` if there are no\nelements.\n* If any of elements is `NaN` returns `NaN`.
```

```
*\n@SinceKotlin("1.4")\npublic fun\nIterable<Double>.maxOrNull(): Double? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var\n    max = iterator.next()\n    while (iterator.hasNext()) {\n        val e = iterator.next()\n        max = maxOf(max, e)\n    }\n    return max\n}\n\n* Returns the largest element or `null` if there are no elements.\n* If any of\nelements is `NaN` returns `NaN`.
```

```
*\n@SinceKotlin("1.4")\npublic fun Iterable<Float>.maxOrNull(): Float? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var\n    max = iterator.next()\n    while (iterator.hasNext()) {\n        val e = iterator.next()\n        max = maxOf(max, e)\n    }\n    return max\n}\n\n* Returns the largest element or `null` if there are no elements.
```

```
*\n@SinceKotlin("1.4")\npublic fun <T : Comparable<T>> Iterable<T>.maxOrNull(): T? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var\n    max = iterator.next()\n    while (iterator.hasNext()) {\n        val e = iterator.next()\n        if (max < e) max = e\n    }\n    return max\n}\n\n* Returns the largest element or `null` if there are no elements.
```

```

}
return max
}
@Deprecated("Use maxWithOrNull instead.",
ReplaceWith("this.maxWithOrNull(comparator)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun <T> Iterable<T>.maxWith(comparator: Comparator<in T>): T? {
return maxWithOrNull(comparator)
}
/**
Returns the first element having the largest value according to the
provided [comparator] or `null` if there are no elements.
*/
@SinceKotlin("1.4")
public fun <T>
Iterable<T>.maxWithOrNull(comparator: Comparator<in T>): T? {
val iterator = iterator()
if (!iterator.hasNext()) return null
var max = iterator.next()
while (iterator.hasNext()) {
val e =
iterator.next()
if (comparator.compare(max, e) < 0) max = e
}
return max
}
@Deprecated("Use
minOrNull instead.", ReplaceWith("this.minOrNull()"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
@SinceKotlin("1.1")
public fun Iterable<Double>.min(): Double?
{
return minOrNull()
}
@Deprecated("Use minOrNull instead.",
ReplaceWith("this.minOrNull()"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
@SinceKotlin("1.1")
public fun Iterable<Float>.min(): Float? {
return
minOrNull()
}
@Deprecated("Use minOrNull instead.",
ReplaceWith("this.minOrNull()"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public fun <T : Comparable<T>> Iterable<T>.min(): T? {
return
minOrNull()
}
@Deprecated("Use minByOrNull instead.",
ReplaceWith("this.minByOrNull(selector)"))
@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")
public inline fun <T, R : Comparable<R>> Iterable<T>.minBy(selector: (T) -> R):
T? {
return minByOrNull(selector)
}
/**
Returns the first element yielding the smallest value of the
given function or `null` if there are no elements.
*/
@sample
samples.collections.Collections.Aggregates.minByOrNull
*/
@SinceKotlin("1.4")
public inline fun <T, R :
Comparable<R>> Iterable<T>.minByOrNull(selector: (T) -> R): T? {
val iterator = iterator()
if (!iterator.hasNext()) return null
var minElem = iterator.next()
if (!iterator.hasNext()) return minElem
var
minValue = selector(minElem)
do {
val e = iterator.next()
val v = selector(e)
if (minValue >
v) {
minElem = e
minValue = v
}
} while (iterator.hasNext())
return
minElem
}
/**
Returns the smallest value among all values produced by [selector] function
*/
applied to
each element in the collection.
*/
If any of values produced by [selector] function is `NaN`, the returned result
is `NaN`.
*/
@throws NoSuchElementException if the collection is empty.
}
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@kotlin.internal.InlineOnly
public inline fun <T> Iterable<T>.minOf(selector: (T) ->
Double): Double {
val iterator = iterator()
if (!iterator.hasNext()) throw NoSuchElementException()
var
minValue = selector(iterator.next())
while (iterator.hasNext()) {
val v = selector(iterator.next())
minValue = minOf(minValue, v)
}
return minValue
}
/**
Returns the smallest value among all
values produced by [selector] function
*/
applied to each element in the collection.
*/
If any of values
produced by [selector] function is `NaN`, the returned result is `NaN`.
*/
@throws NoSuchElementException
if the collection is empty.
}
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@kotlin.internal.InlineOnly
public inline fun <T> Iterable<T>.minOf(selector: (T) ->
Float): Float {
val iterator = iterator()
if (!iterator.hasNext()) throw NoSuchElementException()
var
minValue = selector(iterator.next())
while (iterator.hasNext()) {
val v = selector(iterator.next())
minValue = minOf(minValue, v)
}
return minValue
}
/**
Returns the smallest value among all
values produced by [selector] function
*/
applied to each element in the collection.
*/
@throws
NoSuchElementException if the collection is empty.
}
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@kotlin.internal.InlineOnly
public inline fun <T, R : Comparable<R>>
Iterable<T>.minOf(selector: (T) -> R): R {
val iterator = iterator()
if (!iterator.hasNext()) throw
NoSuchElementException()
var minValue = selector(iterator.next())
while (iterator.hasNext()) {
val v

```

```

= selector(iterator.next())\n    if (minValue > v) {\n        minValue = v\n    }\n    }\n    return
minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the collection or `null` if there are no elements.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Iterable<T>.minOrNull(selector: (T)
-> Double): Double? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var minValue =
selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v = selector(iterator.next())\n        minValue =
minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the collection or `null` if there are no elements.\n * \n * If any
of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Iterable<T>.minOrNull(selector: (T)
-> Float): Float? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var minValue =
selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v = selector(iterator.next())\n        minValue =
minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the collection or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Comparable<R>>
Iterable<T>.minOrNull(selector: (T) -> R): R? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return
null\n    var minValue = selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v =
selector(iterator.next())\n        if (minValue > v) {\n            minValue = v\n        }\n    }\n    return
minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values
produced by [selector] function applied to each element in the collection.\n * \n * @throws
NoSuchElementException if the collection is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R> Iterable<T>.minWith(comparator:
Comparator<in R>, selector: (T) -> R): R {\n    val iterator = iterator()\n    if (!iterator.hasNext()) throw
NoSuchElementException()\n    var minValue = selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v
= selector(iterator.next())\n        if (comparator.compare(minValue, v) > 0) {\n            minValue = v\n        }\n    }\n    return
minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all
values produced by [selector] function applied to each element in the collection or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R>
Iterable<T>.minWithOrNull(comparator: Comparator<in R>, selector: (T) -> R): R? {\n    val iterator =
iterator()\n    if (!iterator.hasNext()) return null\n    var minValue = selector(iterator.next())\n    while
(iterator.hasNext()) {\n        val v = selector(iterator.next())\n        if (comparator.compare(minValue, v) > 0) {\n
            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest element or `null` if there are
no elements.\n * \n * If any of elements is `NaN` returns `NaN`.\n *\n@SinceKotlin("1.4")\npublic fun
Iterable<Double>.minOrNull(): Double? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var
min = iterator.next()\n    while (iterator.hasNext()) {\n        val e = iterator.next()\n        min = minOf(min, e)\n    }\n    return
min\n}\n\n/**\n * Returns the smallest element or `null` if there are no elements.\n * \n * If any of elements
is `NaN` returns `NaN`.\n *\n@SinceKotlin("1.4")\npublic fun Iterable<Float>.minOrNull(): Float? {\n    val
iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var min = iterator.next()\n    while (iterator.hasNext())
{\n        val e = iterator.next()\n        min = minOf(min, e)\n    }\n    return min\n}\n\n/**\n * Returns the smallest
element or `null` if there are no elements.\n *\n@SinceKotlin("1.4")\npublic fun <T : Comparable<T>>
Iterable<T>.minOrNull(): T? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var min =
iterator.next()\n    while (iterator.hasNext()) {\n        val e = iterator.next()\n        if (min > e) min = e\n    }\n    return

```

```

min\n}\n\n@Deprecated(\n"Use minWithOrNull instead.\n",
ReplaceWith(\n"this.minWithOrNull(comparator)\n"))\n\n@DeprecatedSinceKotlin(warningSince = \n"1.4\n", errorSince
= \n"1.5\n", hiddenSince = \n"1.6\n")\n\npublic fun <T> Iterable<T>.minWith(comparator: Comparator<in T>): T? {\n
return minWithOrNull(comparator)\n}\n\n/**\n * Returns the first element having the smallest value according to
the provided [comparator] or `null` if there are no elements.\n * \n\n@SinceKotlin(\n"1.4\n")\n\npublic fun <T>
Iterable<T>.minWithOrNull(comparator: Comparator<in T>): T? {\n    val iterator = iterator()\n    if
(!iterator.hasNext()) return null\n    var min = iterator.next()\n    while (iterator.hasNext()) {\n        val e =
iterator.next()\n        if (comparator.compare(min, e) > 0) min = e\n    }\n    return min\n}\n\n/**\n * Returns `true` if
the collection has no elements.\n * \n\n * @sample samples.collections.Collections.Aggregates.none\n * \n\npublic fun
<T> Iterable<T>.none(): Boolean {\n    if (this is Collection) return isEmpty()\n    return
!iterator().hasNext()\n}\n\n/**\n * Returns `true` if no elements match the given [predicate].\n * \n\n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n * \n\npublic inline fun <T>
Iterable<T>.none(predicate: (T) -> Boolean): Boolean {\n    if (this is Collection && isEmpty()) return true\n    for
(element in this) if (predicate(element)) return false\n    return true\n}\n\n/**\n * Performs the given [action] on each
element and returns the collection itself afterwards.\n * \n\n@SinceKotlin(\n"1.1\n")\n\npublic inline fun <T, C :
Iterable<T>> C.onEach(action: (T) -> Unit): C {\n    return apply { for (element in this) action(element)
}\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index with the element,\n * and
returns the collection itself afterwards.\n * \n\n * @param [action] function that takes the index of an element and the
element itself\n * and performs the action on the element.\n * \n\n@SinceKotlin(\n"1.4\n")\n\npublic inline fun <T, C :
Iterable<T>> C.onEachIndexed(action: (index: Int, T) -> Unit): C {\n    return apply { forEachIndexed(action)
}\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from left to right\n * to
current accumulator value and each element.\n * \n\n * Throws an exception if this collection is empty. If the
collection can be empty in an expected way,\n * please use [reduceOrNull] instead. It returns `null` when its receiver
is empty.\n * \n\n * @param [operation] function that takes current accumulator value and an element,\n * and
calculates the next accumulator value.\n * \n\n * @sample samples.collections.Collections.Aggregates.reduce\n * \n\npublic inline fun <S, T : S> Iterable<T>.reduce(operation: (acc: S, T) -> S): S {\n    val iterator = this.iterator()\n
    if (!iterator.hasNext()) throw UnsupportedOperationException(\n"Empty collection can't be reduced.\n")\n    var
accumulator: S = iterator.next()\n    while (iterator.hasNext()) {\n        accumulator = operation(accumulator,
iterator.next())\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the first element and
applying [operation] from left to right\n * to current accumulator value and each element with its index in the
original collection.\n * \n\n * Throws an exception if this collection is empty. If the collection can be empty in an
expected way,\n * please use [reduceIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n\n *
@param [operation] function that takes the index of an element, current accumulator value and the element itself,\n *
and calculates the next accumulator value.\n * \n\n * @sample samples.collections.Collections.Aggregates.reduce\n * \n\npublic inline fun <S, T : S> Iterable<T>.reduceIndexed(operation: (index: Int, acc: S, T) -> S): S {\n    val
iterator = this.iterator()\n    if (!iterator.hasNext()) throw UnsupportedOperationException(\n"Empty collection can't
be reduced.\n")\n    var index = 1\n    var accumulator: S = iterator.next()\n    while (iterator.hasNext()) {\n
accumulator = operation(checkIndexOverflow(index++), accumulator, iterator.next())\n    }\n    return
accumulator\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from left to
right\n * to current accumulator value and each element with its index in the original collection.\n * \n\n * Returns
`null` if the collection is empty.\n * \n\n * @param [operation] function that takes the index of an element, current
accumulator value and the element itself,\n * and calculates the next accumulator value.\n * \n\n * @sample
samples.collections.Collections.Aggregates.reduceOrNull\n * \n\n@SinceKotlin(\n"1.4\n")\n\npublic inline fun <S, T : S>
Iterable<T>.reduceIndexedOrNull(operation: (index: Int, acc: S, T) -> S): S? {\n    val iterator = this.iterator()\n
    if (!iterator.hasNext()) return null\n    var index = 1\n    var accumulator: S = iterator.next()\n    while
(iterator.hasNext()) {\n        accumulator = operation(checkIndexOverflow(index++), accumulator, iterator.next())\n
    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation]
from left to right\n * to current accumulator value and each element.\n * \n\n * Returns `null` if the collection is

```

```

empty.\n * \n * @param [operation] function that takes current accumulator value and an element,\n * and calculates
the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun <S, T : S>
Iterable<T>.reduceOrNull(operation: (acc: S, T) -> S): S? {\n    val iterator = this.iterator()\n    if
(!iterator.hasNext()) return null\n    var accumulator: S = iterator.next()\n    while (iterator.hasNext()) {\n
accumulator = operation(accumulator, iterator.next())\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value
starting with the last element and applying [operation] from right to left\n * to each element and current accumulator
value.\n * \n * Throws an exception if this list is empty. If the list can be empty in an expected way,\n * please use
[reduceRightOrNull] instead. It returns `null` when its receiver is empty.\n * \n * @param [operation] function that
takes an element and current accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n
*\npublic inline fun <S, T : S>
List<T>.reduceRight(operation: (T, acc: S) -> S): S {\n    val iterator = listIterator(size)\n    if
(!iterator.hasPrevious())\n        throw UnsupportedOperationException("Empty list can't be reduced.")\n    var
accumulator: S = iterator.previous()\n    while (iterator.hasPrevious()) {\n        accumulator =
operation(iterator.previous(), accumulator)\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting
with the last element and applying [operation] from right to left\n * to each element with its index in the original list
and current accumulator value.\n * \n * Throws an exception if this list is empty. If the list can be empty in an
expected way,\n * please use [reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n
*\n * @param [operation] function that takes the index of an element, the element itself and current accumulator
value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n
*\npublic inline fun <S, T : S>
List<T>.reduceRightIndexed(operation: (index: Int, T, acc: S) -> S): S {\n    val iterator = listIterator(size)\n    if
(!iterator.hasPrevious())\n        throw UnsupportedOperationException("Empty list can't be reduced.")\n    var
accumulator: S = iterator.previous()\n    while (iterator.hasPrevious()) {\n        val index = iterator.previousIndex()\n
accumulator = operation(index, iterator.previous(), accumulator)\n    }\n    return accumulator\n}\n\n/**\n *
Accumulates value starting with the last element and applying [operation] from right to left\n * to each element with
its index in the original list and current accumulator value.\n * \n * Returns `null` if the list is empty.\n * \n *
@param [operation] function that takes the index of an element, the element itself and current accumulator value,\n
* and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\npublic inline fun <S,
T : S> List<T>.reduceRightIndexedOrNull(operation: (index: Int, T, acc: S) -> S): S? {\n    val iterator =
listIterator(size)\n    if (!iterator.hasPrevious())\n        return null\n    var accumulator: S = iterator.previous()\n
while (iterator.hasPrevious()) {\n        val index = iterator.previousIndex()\n        accumulator = operation(index,
iterator.previous(), accumulator)\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the last
element and applying [operation] from right to left\n * to each element and current accumulator value.\n * \n *
Returns `null` if the list is empty.\n * \n * @param [operation] function that takes an element and current
accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun <S, T : S>
List<T>.reduceRightOrNull(operation: (T, acc: S) -> S): S? {\n    val iterator = listIterator(size)\n    if
(!iterator.hasPrevious())\n        return null\n    var accumulator: S = iterator.previous()\n    while
(iterator.hasPrevious()) {\n        accumulator = operation(iterator.previous(), accumulator)\n    }\n    return
accumulator\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying
[operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n
*\n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the
previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an
element, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningFold\n
*\n@SinceKotlin("1.4")\npublic inline fun <T, R>

```

```

Iterable<T>.runningFold(initial: R, operation: (acc: R, T) -> R): List<R> {
    val estimatedSize = collectionSizeOrDefault(9)
    if (estimatedSize == 0) return listOf(initial)
    val result = ArrayList<R>(estimatedSize + 1).apply { add(initial) }
    var accumulator = initial
    for (element in this) {
        accumulator = operation(accumulator, element)
        result.add(accumulator)
    }
    return result
}

Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original collection and current accumulator value that starts with [initial] value.

Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.

@param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.
@sample
samples.collections.Collections.Aggregates.runningFold

^@SinceKotlin("1.4")
public inline fun <T, R>
Iterable<T>.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, T) -> R): List<R> {
    val estimatedSize = collectionSizeOrDefault(9)
    if (estimatedSize == 0) return listOf(initial)
    val result = ArrayList<R>(estimatedSize + 1).apply { add(initial) }
    var index = 0
    var accumulator = initial
    for (element in this) {
        accumulator = operation(index++, accumulator, element)
        result.add(accumulator)
    }
    return result
}

Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with the first element of this collection.

Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.

@param [operation] function that takes current accumulator value and the element, and calculates the next accumulator value.
@sample
samples.collections.Collections.Aggregates.runningReduce

^@SinceKotlin("1.4")
^@WasExperimental(ExperimentalStdlibApi::class)
public inline fun <S, T : S>
Iterable<T>.runningReduce(operation: (acc: S, T) -> S): List<S> {
    val iterator = this.iterator()
    if (!iterator.hasNext()) return emptyList()
    var accumulator: S = iterator.next()
    val result = ArrayList<S>(collectionSizeOrDefault(10)).apply { add(accumulator) }
    while (iterator.hasNext()) {
        accumulator = operation(accumulator, iterator.next())
        result.add(accumulator)
    }
    return result
}

Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original collection and current accumulator value that starts with the first element of this collection.

Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.

@param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.
@sample
samples.collections.Collections.Aggregates.runningReduce

^@SinceKotlin("1.4")
public inline fun <S, T : S>
Iterable<T>.runningReduceIndexed(operation: (index: Int, acc: S, T) -> S): List<S> {
    val iterator = this.iterator()
    if (!iterator.hasNext()) return emptyList()
    var accumulator: S = iterator.next()
    val result = ArrayList<S>(collectionSizeOrDefault(10)).apply {
        add(accumulator)
    }
    var index = 1
    while (iterator.hasNext()) {
        accumulator = operation(index++, accumulator, iterator.next())
        result.add(accumulator)
    }
    return result
}

Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with [initial] value.

Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.

@param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.
@sample
samples.collections.Collections.Aggregates.scan

^@SinceKotlin("1.4")
^@WasExperimental(ExperimentalStdlibApi::class)
public inline fun <T, R>
Iterable<T>.scan(initial: R, operation: (acc: R, T) -> R): List<R> {
    return runningFold(initial, operation)
}

Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original collection and current accumulator value that starts with [initial] value.

Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.

@param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator

```

```

value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun <T, R>
Iterable<T>.scanIndexed(initial: R, operation: (index: Int, acc: R, T) -> R): List<R> {\n return
runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the collection.\n *\n@Deprecated("Use sumOf instead.\",
ReplaceWith("this.sumOf(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun <T>
Iterable<T>.sumBy(selector: (T) -> Int): Int {\n var sum: Int = 0\n for (element in this) {\n sum +=
selector(element)\n }\n return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the collection.\n *\n@Deprecated("Use sumOf instead.\",
ReplaceWith("this.sumOf(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun <T>
Iterable<T>.sumByDouble(selector: (T) -> Double): Double {\n var sum: Double = 0.0\n for (element in this)
{\n sum += selector(element)\n }\n return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the collection.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfDouble")\n@kotlin.internal.InlineOnly\npublic inline fun
<T> Iterable<T>.sumOf(selector: (T) -> Double): Double {\n var sum: Double = 0.toDouble()\n for (element in
this) {\n sum += selector(element)\n }\n return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the collection.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfInt")\n@kotlin.internal.InlineOnly\npublic inline fun <T>
Iterable<T>.sumOf(selector: (T) -> Int): Int {\n var sum: Int = 0.toInt()\n for (element in this) {\n sum +=
selector(element)\n }\n return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the collection.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfLong")\n@kotlin.internal.InlineOnly\npublic inline fun
<T> Iterable<T>.sumOf(selector: (T) -> Long): Long {\n var sum: Long = 0.toLong()\n for (element in this) {\n
sum += selector(element)\n }\n return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the collection.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfUInt")\n@WasExperimental(ExperimentalUnsignedType
s::class)\n@kotlin.internal.InlineOnly\npublic inline fun <T> Iterable<T>.sumOf(selector: (T) -> UInt): UInt {\n
var sum: UInt = 0.toUInt()\n for (element in this) {\n sum += selector(element)\n }\n return
sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the
collection.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedTy
pes::class)\n@kotlin.internal.InlineOnly\npublic inline fun <T> Iterable<T>.sumOf(selector: (T) -> ULong): ULong
{\n var sum: ULong = 0.toULong()\n for (element in this) {\n sum += selector(element)\n }\n return
sum\n}\n\n/**\n * Returns an original collection containing all the non-`null` elements, throwing an
[IllegalArgumentException] if there are any `null` elements.\n *\npublic fun <T : Any>
Iterable<T?>.requireNotNulls(): Iterable<T> {\n for (element in this) {\n if (element == null) {\n throw
IllegalArgumentException("null element found in $this.\")\n }\n }\n
@Suppress("UNCHECKED_CAST")\n return this as Iterable<T>\n}\n\n/**\n * Returns an original collection
containing all the non-`null` elements, throwing an [IllegalArgumentException] if there are any `null` elements.\n
*\npublic fun <T : Any> List<T?>.requireNotNulls(): List<T> {\n for (element in this) {\n if (element ==
null) {\n throw IllegalArgumentException("null element found in $this.\")\n }\n }\n
@Suppress("UNCHECKED_CAST")\n return this as List<T>\n}\n\n/**\n * Splits this collection into a list of
lists each not exceeding the given [size].\n *\n * The last list in the resulting list may have fewer elements than the

```

given [size].\n \* \n \* @param size the number of elements to take in each list, must be positive and can be greater than the number of elements in this collection.\n \* \n \* @sample

```

samples.collections.Collections.Transformations.chunked\n * \n * @SinceKotlin("1.2")\npublic fun <T>
Iterable<T>.chunked(size: Int): List<List<T>> {\n    return windowed(size, size, partialWindows = true)\n}\n\n/**\n * Splits this collection into several lists each not exceeding the given [size]\n * and applies the given [transform] function to an each.\n * \n * @return list of results of the [transform] applied to an each list.\n * \n * Note that the list passed to the [transform] function is ephemeral and is valid only inside that function.\n * You should not store it or allow it to escape in some way, unless you made a snapshot of it.\n * The last list may have fewer elements than the given [size].\n * \n * @param size the number of elements to take in each list, must be positive and can be greater than the number of elements in this collection.\n * \n * @sample samples.text.Strings.chunkedTransform\n * \n * @SinceKotlin("1.2")\npublic fun <T, R> Iterable<T>.chunked(size: Int, transform: (List<T>) -> R): List<R>
{\n    return windowed(size, size, partialWindows = true, transform = transform)\n}\n\n/**\n * Returns a list containing all elements of the original collection without the first occurrence of the given [element].\n * \n * @public operator fun <T> Iterable<T>.minus(element: T): List<T> {\n    val result =
ArrayList<T>(collectionSizeOrDefault(10))\n    var removed = false\n    return this.filterTo(result) { if (!removed && it == element) { removed = true; false } else true }\n}\n\n/**\n * Returns a list containing all elements of the original collection except the elements contained in the given [elements] array.\n * \n * Before Kotlin 1.6, the [elements] array may have been converted to a [HashSet] to speed up the operation, thus the elements were required to have\n * a correct and stable implementation of `hashCode()` that didn't change between successive invocations.\n * On JVM, you can enable this behavior back with the system property
`kotlin.collections.convert_arg_to_set_in_removeAll` set to `true`.\n * \n * @public operator fun <T>
Iterable<T>.minus(elements: Array<out T>): List<T> {\n    if (elements.isEmpty()) return this.toList()\n    val other = elements.convertToSetForSetOperation()\n    return this.filterNot { it in other }\n}\n\n/**\n * Returns a list containing all elements of the original collection except the elements contained in the given [elements] collection.\n * \n * Before Kotlin 1.6, the [elements] collection may have been converted to a [HashSet] to speed up the operation, thus the elements were required to have\n * a correct and stable implementation of `hashCode()` that didn't change between successive invocations.\n * On JVM, you can enable this behavior back with the system property
`kotlin.collections.convert_arg_to_set_in_removeAll` set to `true`.\n * \n * @public operator fun <T>
Iterable<T>.minus(elements: Iterable<T>): List<T> {\n    val other =
elements.convertToSetForSetOperationWith(this)\n    if (other.isEmpty())\n        return this.toList()\n    return this.filterNot { it in other }\n}\n\n/**\n * Returns a list containing all elements of the original collection except the elements contained in the given [elements] sequence.\n * \n * Before Kotlin 1.6, the [elements] sequence may have been converted to a [HashSet] to speed up the operation, thus the elements were required to have\n * a correct and stable implementation of `hashCode()` that didn't change between successive invocations.\n * On JVM, you can enable this behavior back with the system property
`kotlin.collections.convert_arg_to_set_in_removeAll` set to `true`.\n * \n * @public operator fun <T> Iterable<T>.minus(elements: Sequence<T>): List<T> {\n    val other =
elements.convertToSetForSetOperation()\n    if (other.isEmpty())\n        return this.toList()\n    return this.filterNot { it in other }\n}\n\n/**\n * Returns a list containing all elements of the original collection without the first occurrence of the given [element].\n * \n * @kotl
in.internal.InlineOnly\npublic inline fun <T>
Iterable<T>.minusElement(element: T): List<T> {\n    return minus(element)\n}\n\n/**\n * Splits the original collection into pair of lists,\n * where *first* list contains elements for which [predicate] yielded `true`,\n * while *second* list contains elements for which [predicate] yielded `false`.\n * \n * @sample
samples.collections.Iterables.Operations.partition\n * \n * @public inline fun <T> Iterable<T>.partition(predicate: (T) -> Boolean): Pair<List<T>, List<T>> {\n    val first = ArrayList<T>()\n    val second = ArrayList<T>()\n    for (element in this) {\n        if (predicate(element)) {\n            first.add(element)\n        } else {\n            second.add(element)\n        }\n    }\n    return Pair(first, second)\n}\n\n/**\n * Returns a list containing all elements of the original collection and then the given [element].\n * \n * @public operator fun <T> Iterable<T>.plus(element: T): List<T> {\n    if (this is Collection) return this.plus(element)\n    val result = ArrayList<T>()\n
```

```

result.addAll(this)\n result.add(element)\n return result\n}\n\n/**\n * Returns a list containing all elements of the
original collection and then the given [element].\n *\npublic operator fun <T> Collection<T>.plus(element: T):
List<T> {\n    val result = ArrayList<T>(size + 1)\n    result.addAll(this)\n    result.add(element)\n    return
result\n}\n\n/**\n * Returns a list containing all elements of the original collection and then all elements of the given
[elements] array.\n *\npublic operator fun <T> Iterable<T>.plus(elements: Array<out T>): List<T> {\n    if (this is
Collection) return this.plus(elements)\n    val result = ArrayList<T>()\n    result.addAll(this)\n
result.addAll(elements)\n    return result\n}\n\n/**\n * Returns a list containing all elements of the original
collection and then all elements of the given [elements] array.\n *\npublic operator fun <T>
Collection<T>.plus(elements: Array<out T>): List<T> {\n    val result = ArrayList<T>(this.size + elements.size)\n
result.addAll(this)\n    result.addAll(elements)\n    return result\n}\n\n/**\n * Returns a list containing all elements
of the original collection and then all elements of the given [elements] collection.\n *\npublic operator fun <T>
Iterable<T>.plus(elements: Iterable<T>): List<T> {\n    if (this is Collection) return this.plus(elements)\n    val
result = ArrayList<T>()\n    result.addAll(this)\n    result.addAll(elements)\n    return result\n}\n\n/**\n * Returns a
list containing all elements of the original collection and then all elements of the given [elements] collection.\n\n
*\npublic operator fun <T> Collection<T>.plus(elements: Iterable<T>): List<T> {\n    if (elements is Collection)
{\n        val result = ArrayList<T>(this.size + elements.size)\n        result.addAll(this)\n
result.addAll(elements)\n        return result\n    } else {\n        val result = ArrayList<T>(this)\n
result.addAll(elements)\n        return result\n    }\n}\n\n/**\n * Returns a list containing all elements of the original
collection and then all elements of the given [elements] sequence.\n *\npublic operator fun <T>
Iterable<T>.plus(elements: Sequence<T>): List<T> {\n    val result = ArrayList<T>()\n    result.addAll(this)\n
result.addAll(elements)\n    return result\n}\n\n/**\n * Returns a list containing all elements of the original
collection and then all elements of the given [elements] sequence.\n *\npublic operator fun <T>
Collection<T>.plus(elements: Sequence<T>): List<T> {\n    val result = ArrayList<T>(this.size + 10)\n
result.addAll(this)\n    result.addAll(elements)\n    return result\n}\n\n/**\n * Returns a list containing all elements
of the original collection and then the given [element].\n *\n@kotlin.internal.InlineOnly\npublic inline fun <T>
Iterable<T>.plusElement(element: T): List<T> {\n    return plus(element)\n}\n\n/**\n * Returns a list containing all
elements of the original collection and then the given [element].\n *\n@kotlin.internal.InlineOnly\npublic inline fun
<T> Collection<T>.plusElement(element: T): List<T> {\n    return plus(element)\n}\n\n/**\n * Returns a list of
snapshots of the window of the given [size]\n * sliding along this collection with the given [step], where each\n *
snapshot is a list.\n * \n * Several last lists may have fewer elements than the given [size].\n * \n * Both [size] and
[step] must be positive and can be greater than the number of elements in this collection.\n * @param size the
number of elements to take in each window\n * @param step the number of elements to move the window forward
by on an each step, by default 1\n * @param partialWindows controls whether or not to keep partial windows in the
end if any,\n * by default `false` which means partial windows won't be preserved\n * \n * @sample
samples.collections.Sequences.Transformations.takeWindows\n *\n@SinceKotlin("1.2")\npublic fun <T>
Iterable<T>.windowed(size: Int, step: Int = 1, partialWindows: Boolean = false): List<List<T>> {\n
checkWindowSizeStep(size, step)\n    if (this is RandomAccess && this is List) {\n        val thisSize = this.size\n
val resultCapacity = thisSize / step + if (thisSize % step == 0) 0 else 1\n        val result =
ArrayList<List<T>>(resultCapacity)\n        var index = 0\n        while (index in 0 until thisSize) {\n            val
windowSize = size.coerceAtMost(thisSize - index)\n            if (windowSize < size && !partialWindows) break\n
            result.add(List(windowSize) { this[it + index] })\n            index += step\n        }\n        return result\n    }\n    val
result = ArrayList<List<T>>()\n    windowedIterator(iterator(), size, step, partialWindows, reuseBuffer =
false).forEach {\n        result.add(it)\n    }\n    return result\n}\n\n/**\n * Returns a list of results of applying the
given [transform] function to\n * an each list representing a view over the window of the given [size]\n * sliding
along this collection with the given [step].\n * \n * Note that the list passed to the [transform] function is ephemeral
and is valid only inside that function.\n * You should not store it or allow it to escape in some way, unless you made
a snapshot of it.\n * Several last lists may have fewer elements than the given [size].\n * \n * Both [size] and [step]
must be positive and can be greater than the number of elements in this collection.\n * @param size the number of

```

```

elements to take in each window\n * @param step the number of elements to move the window forward by on an
each step, by default 1\n * @param partialWindows controls whether or not to keep partial windows in the end if
any,\n * by default `false` which means partial windows won't be preserved\n * \n * @sample
samples.collections.Sequences.Transformations.averageWindows\n * \n @SinceKotlin("1.2")\n\npublic fun <T, R>
Iterable<T>.windowed(size: Int, step: Int = 1, partialWindows: Boolean = false, transform: (List<T>) -> R):
List<R> {\n    checkWindowSizeStep(size, step)\n    if (this is RandomAccess && this is List) {\n        val thisSize =
this.size\n        val resultCapacity = thisSize / step + if (thisSize % step == 0) 0 else 1\n        val result =
ArrayList<R>(resultCapacity)\n        val window = MovingSubList(this)\n        var index = 0\n        while (index in 0
until thisSize) {\n            val windowSize = size.coerceAtMost(thisSize - index)\n            if (!partialWindows &&
windowSize < size) break\n            window.move(index, index + windowSize)\n            result.add(transform(window))\n            index += step\n        }\n        return result\n    }\n    val result =
ArrayList<R>()\n    windowedIterator(iterator(), size, step, partialWindows, reuseBuffer = true).forEach {\n
result.add(transform(it))\n    }\n    return result\n}\n\n/**\n * Returns a list of pairs built from the elements of `this`
collection and the [other] array with the same index.\n * The returned list has length of the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n * \n\npublic infix fun <T, R> Iterable<T>.zip(other:
Array<out R>): List<Pair<T, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of values
built from the elements of `this` collection and the [other] array with the same index\n * using the provided
[transform] function applied to each pair of elements.\n * The returned list has length of the shortest collection.\n *
\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n * \n\npublic inline fun <T, R, V>
Iterable<T>.zip(other: Array<out R>, transform: (a: T, b: R) -> V): List<V> {\n    val arraySize = other.size\n    val
list = ArrayList<V>(minOf(collectionSizeOrDefault(10), arraySize))\n    var i = 0\n    for (element in this) {\n        if
(i >= arraySize) break\n        list.add(transform(element, other[i++]))\n    }\n    return list\n}\n\n/**\n * Returns a list
of pairs built from the elements of `this` collection and [other] collection with the same index.\n * The returned list
has length of the shortest collection.\n * \n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n *
\n\npublic infix fun <T, R> Iterable<T>.zip(other: Iterable<R>): List<Pair<T, R>> {\n    return zip(other) { t1, t2 ->
t1 to t2 }\n}\n\n/**\n * Returns a list of values built from the elements of `this` collection and the [other] collection
with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned
list has length of the shortest collection.\n * \n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n * \n\npublic inline fun <T, R, V>
Iterable<T>.zip(other: Iterable<R>, transform: (a: T, b: R) -> V): List<V> {\n    val first = iterator()\n    val second
= other.iterator()\n    val list = ArrayList<V>(minOf(collectionSizeOrDefault(10),
other.collectionSizeOrDefault(10)))\n    while (first.hasNext() && second.hasNext()) {\n
list.add(transform(first.next(), second.next()))\n    }\n    return list\n}\n\n/**\n * Returns a list of pairs of each two
adjacent elements in this collection.\n * \n * \n * The returned list is empty if this collection contains less than two
elements.\n * \n * \n * @sample samples.collections.Collections.Transformations.zipWithNext\n *
\n\n@SinceKotlin("1.2")\n\npublic fun <T> Iterable<T>.zipWithNext(): List<Pair<T, T>> {\n    return zipWithNext
{ a, b -> a to b }\n}\n\n/**\n * Returns a list containing the results of applying the given [transform] function\n * to
an each pair of two adjacent elements in this collection.\n * \n * \n * The returned list is empty if this collection contains
less than two elements.\n * \n * \n * @sample
samples.collections.Collections.Transformations.zipWithNextToFindDeltas\n * \n\n@SinceKotlin("1.2")\n\npublic
inline fun <T, R> Iterable<T>.zipWithNext(transform: (a: T, b: T) -> R): List<R> {\n    val iterator = iterator()\n    if
(!iterator.hasNext()) return emptyList()\n    val result = mutableListOf<R>()\n    var current = iterator.next()\n
while (iterator.hasNext()) {\n        val next = iterator.next()\n        result.add(transform(current, next))\n        current
= next\n    }\n    return result\n}\n\n/**\n * Appends the string from all the elements separated using [separator] and
using the given [prefix] and [postfix] if supplied.\n * \n * \n * If the collection could be huge, you can specify a non-
negative value of [limit], in which case only the first [limit]\n * elements will be appended, followed by the
[truncated] string (which defaults to `"...`).\n * \n * \n * @sample
samples.collections.Collections.Transformations.joinTo\n * \n\npublic fun <T, A : Appendable>

```

```

Iterable<T>.joinTo(buffer: A, separator: CharSequence = '\', '\', prefix: CharSequence = \"\", postfix: CharSequence = \"\", limit: Int = -1, truncated: CharSequence = \"...\", transform: ((T) -> CharSequence)? = null): A {
    buffer.append(prefix)
    var count = 0
    for (element in this) {
        if (++count > 1) buffer.append(separator)
        if (limit < 0 || count <= limit) {
            buffer.appendElement(element, transform)
        } else break
    }
    if (limit >= 0 && count > limit) buffer.append(truncated)
    buffer.append(postfix)
    return buffer
}

Creates a string from all the elements separated using [separator] and using the given [prefix] and [postfix] if supplied.
If the collection could be huge, you can specify a non-negative value of [limit], in which case only the first [limit] elements will be appended, followed by the [truncated] string (which defaults to \"...\").

@sample samples.collections.Collections.Transformations.joinToString

public fun <T>
Iterable<T>.joinToString(separator: CharSequence = '\', '\', prefix: CharSequence = \"\", postfix: CharSequence = \"\", limit: Int = -1, truncated: CharSequence = \"...\", transform: ((T) -> CharSequence)? = null): String {
    return joinTo(StringBuilder(), separator, prefix, postfix, limit, truncated, transform).toString()
}

Returns this collection as an [Iterable].

@kotlin.internal.InlineOnly
public inline fun <T> Iterable<T>.asIterable():
Iterable<T> {
    return this
}

Creates a [Sequence] instance that wraps the original collection returning its elements when being iterated.

@sample
samples.collections.Sequences.Building.sequenceFromCollection

public fun <T> Iterable<T>.asSequence():
Sequence<T> {
    return Sequence { this.iterator() }
}

Returns an average value of elements in the collection.

@kotlin.jvm.JvmName("averageOfByte")
public fun Iterable<Byte>.average(): Double {
    var sum: Double = 0.0
    var count: Int = 0
    for (element in this) {
        sum += element
        checkCountOverflow(++count)
    }
    return if (count == 0) Double.NaN else sum / count
}

Returns an average value of elements in the collection.

@kotlin.jvm.JvmName("averageOfShort")
public fun
Iterable<Short>.average(): Double {
    var sum: Double = 0.0
    var count: Int = 0
    for (element in this) {
        sum += element
        checkCountOverflow(++count)
    }
    return if (count == 0) Double.NaN else sum / count
}

Returns an average value of elements in the collection.

@kotlin.jvm.JvmName("averageOfInt")
public fun Iterable<Int>.average(): Double {
    var sum: Double = 0.0
    var count: Int = 0
    for (element in this) {
        sum += element
        checkCountOverflow(++count)
    }
    return if (count == 0) Double.NaN else sum / count
}

Returns an average value of elements in the collection.

@kotlin.jvm.JvmName("averageOfLong")
public fun Iterable<Long>.average(): Double {
    var sum: Double = 0.0
    var count: Int = 0
    for (element in this) {
        sum += element
        checkCountOverflow(++count)
    }
    return if (count == 0) Double.NaN else sum / count
}

Returns an average value of elements in the collection.

@kotlin.jvm.JvmName("averageOfFloat")
public fun
Iterable<Float>.average(): Double {
    var sum: Double = 0.0
    var count: Int = 0
    for (element in this) {
        sum += element
        checkCountOverflow(++count)
    }
    return if (count == 0) Double.NaN else sum / count
}

Returns an average value of elements in the collection.

@kotlin.jvm.JvmName("averageOfDouble")
public fun Iterable<Double>.average(): Double {
    var sum: Double = 0.0
    var count: Int = 0
    for (element in this) {
        sum += element
        checkCountOverflow(++count)
    }
    return if (count == 0) Double.NaN else sum / count
}

Returns the sum of all elements in the collection.

@kotlin.jvm.JvmName("sumOfByte")
public fun
Iterable<Byte>.sum(): Int {
    var sum: Int = 0
    for (element in this) {
        sum += element
    }
    return sum
}

Returns the sum of all elements in the collection.

@kotlin.jvm.JvmName("sumOfShort")
public fun Iterable<Short>.sum(): Int {
    var sum: Int = 0
    for (element in this) {
        sum += element
    }
    return sum
}

Returns the sum of all elements in the collection.

@kotlin.jvm.JvmName("sumOfInt")
public fun Iterable<Int>.sum(): Int {
    var sum: Int = 0
    for (element in this) {
        sum += element
    }
    return sum
}

Returns the sum of all elements in the collection.

@kotlin.jvm.JvmName("sumOfLong")
public fun Iterable<Long>.sum(): Long {
    var sum: Long = 0L
    for (element in this) {
        sum += element
    }
    return sum
}

Returns the sum of all elements in the collection.

@kotlin.jvm.JvmName("sumOfFloat")
public fun
Iterable<Float>.sum(): Float {
    var sum: Float = 0.0f
    for (element in this) {
        sum += element
    }
}

```

```

return sum\n}\n\n/**\n * Returns the sum of all elements in the collection.\n
*\n@kotlin.jvm.JvmName("sumOfDouble")\npublic fun Iterable<Double>.sum(): Double {\n    var sum: Double
= 0.0\n    for (element in this) {\n        sum += element\n    }\n    return sum\n}\n\n"/\n * Copyright 2010-2018
JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n *\n\npackage kotlin.collections\n\nimport
kotlin.comparisons.naturalOrder\n\nimport kotlin.random.Random\n\n/**\n * Returns the array if it's not `null`, or an
empty array otherwise.\n * @sample samples.collections.Arrays.Usage.arrayOrEmpty\n
*\n@kotlin.internal.InlineOnly\npublic actual inline fun <T> Array<out T>?.orEmpty(): Array<out T> = this ?:
emptyArray<T>()\n\n/**\n * Returns a *typed* array containing all of the elements of this collection.\n *\n *
Allocates an array of runtime type `T` having its size equal to the size of this collection\n * and populates the array
with the elements of this collection.\n * @sample
samples.collections.Collections.Collections.collectionToTypedArray\n *\n@kotlin.internal.InlineOnly\npublic
actual inline fun <T> Collection<T>.toArray(): Array<T> =
toArray(this)\n\n@JsName("toArray")\n@PublishedApi\ninternal fun <T> toArray(collection:
Collection<T>): Array<T> {\n    return if (collection.asDynamic().toArray !== undefined)\n        collection.asDynamic().toArray().unsafeCast<Array<T>>()\n    else\n        toArrayImpl(collection).unsafeCast<Array<T>>()\n}\n\n@JsName("toArrayImpl")\ninternal actual fun
toArrayImpl(collection: Collection<*>): Array<Any?> {\n    val array = emptyArray<Any?>()\n    val iterator
= collection.iterator()\n    while (iterator.hasNext())\n        array.asDynamic().push(iterator.next())\n    return
array\n}\n\n@JsName("copyToExistingArrayImpl")\ninternal actual fun <T> copyToArrayImpl(collection:
Collection<*>, array: Array<T>): Array<T> {\n    if (array.size < collection.size)\n        return
copyToArrayImpl(collection).unsafeCast<Array<T>>()\n    val iterator = collection.iterator()\n    var index = 0\n    while (iterator.hasNext()) {\n        array[index++] = iterator.next().unsafeCast<T>()\n    }\n    if (index < array.size)\n        array[index] = null.unsafeCast<T>()\n    }\n    return array\n}\n\n/**\n * Returns an immutable list
containing only the specified object [element].\n *\npublic fun <T> listOf(element: T): List<T> =
listOf(element)\n\n@PublishedApi\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\ninternal actual
inline fun <E> buildListInternal(builderAction: MutableList<E>().->Unit): List<E> {\n    return
ArrayList<E>().apply(builderAction).build()\n}\n\n@PublishedApi\n@SinceKotlin("1.3")\n@kotlin.internal.Inlin
eOnly\ninternal actual inline fun <E> buildListInternal(capacity: Int, builderAction: MutableList<E>().->Unit):
List<E> {\n    checkBuilderCapacity(capacity)\n    return
ArrayList<E>(capacity).apply(builderAction).build()\n}\n\n/**\n * Returns an immutable set containing only the
specified object [element].\n *\npublic fun <T> setOf(element: T): Set<T> =
hashSetOf(element)\n\n@PublishedApi\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\ninternal actual inline
fun <E> buildSetInternal(builderAction: MutableSet<E>().->Unit): Set<E> {\n    return
LinkedHashSet<E>().apply(builderAction).build()\n}\n\n@PublishedApi\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\ninternal actual inline fun <E> buildSetInternal(capacity: Int, builderAction: MutableSet<E>().->Unit):
Set<E> {\n    return LinkedHashSet<E>(capacity).apply(builderAction).build()\n}\n\n/**\n * Returns an
immutable map, mapping only the specified key to the\n * specified value.\n *\npublic fun <K, V> mapOf(pair:
Pair<K, V>): Map<K, V> =
hashMapOf(pair)\n\n@PublishedApi\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\ninternal actual inline
fun <K, V> buildMapInternal(builderAction: MutableMap<K, V>().->Unit): Map<K, V> {\n    return
LinkedHashMap<K,
V>().apply(builderAction).build()\n}\n\n@PublishedApi\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\ninte
rnal actual inline fun <K, V> buildMapInternal(capacity: Int, builderAction: MutableMap<K, V>().->Unit):
Map<K, V> {\n    return LinkedHashMap<K, V>(capacity).apply(builderAction).build()\n}\n\n/**\n * Fills the
list with the provided [value].\n *\n * Each element in the list gets replaced with the [value].\n
*\n@kotlin.internal.InlineOnly\n@SinceKotlin("1.2")\npublic actual fun <T> MutableList<T>.fill(value: T): Unit {\n    for (index in
0..lastIndex) {\n        this[index] = value\n    }\n}\n\n/**\n * Randomly shuffles elements in this list.\n *\n * See:

```

[https://en.wikipedia.org/wiki/Fisher%E2%80%93Yates\\_shuffle#The\\_modern\\_algorithm](https://en.wikipedia.org/wiki/Fisher%E2%80%93Yates_shuffle#The_modern_algorithm)

```

*\/n@SinceKotlin("1.2")\npublic actual fun <T> MutableList<T>.shuffle(): Unit = shuffle(Random)\n\n/**\n * Returns a new list with the elements of this list randomly shuffled.\n */\n@SinceKotlin("1.2")\npublic actual fun <T> Iterable<T>.shuffled(): List<T> = toMutableList().apply { shuffle() }\n\n/**\n * Sorts elements in the list in-place according to their natural sort order.\n */\n * The sort is _stable_. It means that equal elements preserve their order relative to each other after sorting.\n */\n * @sample samples.collections.Collections.Sorting.sortMutableList\n */\npublic actual fun <T : Comparable<T>> MutableList<T>.sort(): Unit {\n    collectionsSort(this, naturalOrder())\n}\n\n/**\n * Sorts elements in the list in-place according to the order specified with [comparator].\n */\n * The sort is _stable_. It means that equal elements preserve their order relative to each other after sorting.\n */\n * @sample samples.collections.Collections.Sorting.sortMutableListWith\n */\npublic actual fun <T> MutableList<T>.sortWith(comparator: Comparator<in T>): Unit {\n    collectionsSort(this, comparator)\n}\n\nprivate fun <T> collectionsSort(list: MutableList<T>, comparator: Comparator<in T>) {\n    if (list.size <= 1) return\n    val array = copyToArray(list)\n    sortArrayWith(array, comparator)\n    for (i in 0 until array.size) {\n        list[i] = array[i]\n    }\n}\n\ninternal actual fun <T> arrayOfNulls(reference: Array<T>, size: Int): Array<T> {\n    return arrayOfNulls<Any>(size).unsafeCast<Array<T>>()\n}\n\n@SinceKotlin("1.3")\n@PublishedApi\n@JsName("arrayCopy")\ninternal fun <T> arrayCopy(source: Array<out T>, destination: Array<in T>, destinationOffset: Int, startIndex: Int, endIndex: Int) {\n    AbstractList.checkRangeIndexes(startIndex, endIndex, source.size)\n    val rangeSize = endIndex - startIndex\n    AbstractList.checkRangeIndexes(destinationOffset, destinationOffset + rangeSize, destination.size)\n    if (js("ArrayBuffer").isView(destination) && js("ArrayBuffer").isView(source)) {\n        val subrange = source.asDynamic().subarray(startIndex, endIndex)\n        destination.asDynamic().set(subrange, destinationOffset)\n    } else {\n        if (source !== destination || destinationOffset <= startIndex) {\n            for (index in 0 until rangeSize) {\n                destination[destinationOffset + index] = source[startIndex + index]\n            }\n        } else {\n            for (index in rangeSize - 1 downTo 0) {\n                destination[destinationOffset + index] = source[startIndex + index]\n            }\n        }\n    }\n}\n\n// no singleton map implementation in js, return map as is\n@Suppress("NOTHING_TO_INLINE")\ninternal actual inline fun <K, V> Map<K, V>.toSingletonMapOrSelf(): Map<K, V> = this\n\n@Suppress("NOTHING_TO_INLINE")\ninternal actual inline fun <K, V> Map<out K, V>.toSingletonMap(): Map<K, V> = this.toMutableMap()\n\n@Suppress("NOTHING_TO_INLINE")\ninternal actual inline fun <T> Array<out T>.copyToArrayOfAny(isVarargs: Boolean): Array<out Any?> =\n    if (isVarargs)\n        // no need to copy vararg array in JS\n        this\n    else\n        this.copyOf()\n\n@PublishedApi\ninternal actual fun checkIndexOverflow(index: Int): Int {\n    if (index < 0) {\n        throwIndexOverflow()\n    }\n    return index\n}\n\n@PublishedApi\ninternal actual fun checkCountOverflow(count: Int): Int {\n    if (count < 0) {\n        throwCountOverflow()\n    }\n    return count\n}\n\n\n/**\n * JS map and set implementations do not make use of capacities or load factors.\n */\n@PublishedApi\ninternal actual fun mapCapacity(expectedSize: Int) = expectedSize\n\n/**\n * Checks a collection builder function capacity argument.\n */\n * In JS no validation is made in Map/Set constructor yet.\n */\n@SinceKotlin("1.3")\n@PublishedApi\ninternal fun checkBuilderCapacity(capacity: Int) {\n    require(capacity >= 0) { "capacity must be non-negative." }\n}\n\n\ninternal actual fun brittleContainsOptimizationEnabled(): Boolean = false", /*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n */\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("CollectionsKt")\n\npackage kotlin.collections\n\n/**\n * Returns the given iterator itself. This allows to use an instance of iterator in a `for` loop.\n */\n * @sample samples.collections.Iterators.iterator\n */\n@kotlin.internal.InlineOnly\npublic inline operator fun <T> Iterator<T>.iterator(): Iterator<T> = this\n\n/**\n * Returns an [Iterator] that wraps each element produced by the original iterator\n */\n * into an [IndexedValue] containing the index of that element and the element itself.\n */\n * @sample samples.collections.Iterators.withIndexIterator\n */\npublic fun <T> Iterator<T>.withIndex():

```

```

Iterator<IndexedValue<T>> = IndexingIterator(this)\n\n/**\n * Performs the given [operation] on each element of
this [Iterator].\n * @sample samples.collections.Iterators.forEachIterator\n */\npublic inline fun <T>
Iterator<T>.forEach(operation: (T) -> Unit): Unit {\n    for (element in this) operation(element)\n}\n\n/**\n *
Iterator transforming original `iterator` into iterator of [IndexedValue], counting index from zero.\n */\ninternal class
IndexingIterator<out T>(private val iterator: Iterator<T>) : Iterator<IndexedValue<T>> {\n    private var index =
0\n    final override fun hasNext(): Boolean = iterator.hasNext()\n    final override fun next(): IndexedValue<T> =
IndexedValue(checkIndexOverflow(index++), iterator.next())\n}\n\n",*/\n * Copyright 2010-2021 JetBrains s.r.o.
and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license
that can be found in the license/LICENSE.txt file.\n
*/\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("ComparisonsKt")\n\npackage
kotlin.comparisons\n\n/\n// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt\n// See:
https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n/\n\nimport kotlin.random.*\n\n/**\n * Returns the
greater of two values.\n * \n * If values are equal, returns the first one.\n */\n@SinceKotlin("1.1")\npublic expect
fun <T : Comparable<T>> maxOf(a: T, b: T): T\n\n/**\n * Returns the greater of two values.\n
*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun maxOf(a: Byte, b: Byte):
Byte\n\n/**\n * Returns the greater of two values.\n */\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic
expect inline fun maxOf(a: Short, b: Short): Short\n\n/**\n * Returns the greater of two values.\n
*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun maxOf(a: Int, b: Int): Int\n\n/**\n
* Returns the greater of two values.\n */\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline
fun maxOf(a: Long, b: Long): Long\n\n/**\n * Returns the greater of two values.\n * \n * If either value is `NaN`,
returns `NaN`.\n */\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun maxOf(a: Float,
b: Float): Float\n\n/**\n * Returns the greater of two values.\n * \n * If either value is `NaN`, returns `NaN`.\n
*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun maxOf(a: Double, b: Double):
Double\n\n/**\n * Returns the greater of three values.\n * \n * If there are multiple equal maximal values, returns the
first of them.\n */\n@SinceKotlin("1.1")\npublic expect fun <T : Comparable<T>> maxOf(a: T, b: T, c: T):
T\n\n/**\n * Returns the greater of three values.\n */\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic
expect inline fun maxOf(a: Byte, b: Byte, c: Byte): Byte\n\n/**\n * Returns the greater of three values.\n
*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun maxOf(a: Short, b: Short, c:
Short): Short\n\n/**\n * Returns the greater of three values.\n */\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic
expect inline fun maxOf(a: Int, b: Int, c: Int): Int\n\n/**\n * Returns the greater of three values.\n */\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic
expect inline fun maxOf(a: Long, b: Long, c: Long): Long\n\n/**\n * Returns the greater of three values.\n * \n * If
any value is `NaN`, returns `NaN`.\n */\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline
fun maxOf(a: Float, b: Float, c: Float): Float\n\n/**\n * Returns the greater of three values.\n * \n * If any value is
`NaN`, returns `NaN`.\n */\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun maxOf(a:
Double, b: Double, c: Double): Double\n\n/**\n * Returns the greater of three values according to the order
specified by the given [comparator].\n * \n * If there are multiple equal maximal values, returns the first of them.\n
*/\n@SinceKotlin("1.1")\npublic fun <T> maxOf(a: T, b: T, c: T, comparator: Comparator<in T>): T {\n    return
maxOf(a, maxOf(b, c, comparator), comparator)\n}\n\n/**\n * Returns the greater of two values according to the
order specified by the given [comparator].\n * \n * If values are equal, returns the first one.\n
*/\n@SinceKotlin("1.1")\npublic fun <T> maxOf(a: T, b: T, comparator: Comparator<in T>): T {\n    return if
(comparator.compare(a, b) >= 0) a else b\n}\n\n/**\n * Returns the greater of the given values.\n * \n * If there are
multiple equal maximal values, returns the first of them.\n */\n@SinceKotlin("1.4")\npublic expect fun <T :
Comparable<T>> maxOf(a: T, vararg other: T): T\n\n/**\n * Returns the greater of the given values.\n
*/\n@SinceKotlin("1.4")\npublic expect fun maxOf(a: Byte, vararg other: Byte): Byte\n\n/**\n * Returns the
greater of the given values.\n */\n@SinceKotlin("1.4")\npublic expect fun maxOf(a: Short, vararg other: Short):
Short\n\n/**\n * Returns the greater of the given values.\n */\n@SinceKotlin("1.4")\npublic expect fun maxOf(a:
Int, vararg other: Int): Int\n\n/**\n * Returns the greater of the given values.\n */\n@SinceKotlin("1.4")\npublic

```

expect fun maxOf(a: Long, vararg other: Long): Long\n\n/\*\*\n \* Returns the greater of the given values.\n \* \n \* If any value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.4")\npublic expect fun maxOf(a: Float, vararg other: Float): Float\n\n/\*\*\n \* Returns the greater of the given values.\n \* \n \* If any value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.4")\npublic expect fun maxOf(a: Double, vararg other: Double): Double\n\n/\*\*\n \* Returns the greater of the given values according to the order specified by the given [comparator].\n \* \n \* If there are multiple equal maximal values, returns the first of them.\n \*/\n@SinceKotlin("1.4")\npublic fun <T> maxOf(a: T, vararg other: T, comparator: Comparator<in T>): T {\n var max = a\n for (e in other) if (comparator.compare(max, e) < 0) max = e\n return max\n}\n\n/\*\*\n \* Returns the smaller of two values.\n \* \n \* If values are equal, returns the first one.\n \*/\n@SinceKotlin("1.1")\npublic expect fun <T : Comparable<T>> minOf(a: T, b: T): T\n\n/\*\*\n \* Returns the smaller of two values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun minOf(a: Byte, b: Byte): Byte\n\n/\*\*\n \* Returns the smaller of two values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun minOf(a: Short, b: Short): Short\n\n/\*\*\n \* Returns the smaller of two values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun minOf(a: Int, b: Int): Int\n\n/\*\*\n \* Returns the smaller of two values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun minOf(a: Long, b: Long): Long\n\n/\*\*\n \* Returns the smaller of two values.\n \* \n \* If either value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun minOf(a: Float, b: Float): Float\n\n/\*\*\n \* Returns the smaller of two values.\n \* \n \* If either value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun minOf(a: Double, b: Double): Double\n\n/\*\*\n \* Returns the smaller of three values.\n \* \n \* If there are multiple equal minimal values, returns the first of them.\n \*/\n@SinceKotlin("1.1")\npublic expect fun <T : Comparable<T>> minOf(a: T, b: T, c: T): T\n\n/\*\*\n \* Returns the smaller of three values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun minOf(a: Byte, b: Byte, c: Byte): Byte\n\n/\*\*\n \* Returns the smaller of three values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun minOf(a: Short, b: Short, c: Short): Short\n\n/\*\*\n \* Returns the smaller of three values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun minOf(a: Int, b: Int, c: Int): Int\n\n/\*\*\n \* Returns the smaller of three values.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun minOf(a: Long, b: Long, c: Long): Long\n\n/\*\*\n \* Returns the smaller of three values.\n \* \n \* If any value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun minOf(a: Float, b: Float, c: Float): Float\n\n/\*\*\n \* Returns the smaller of three values.\n \* \n \* If any value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic expect inline fun minOf(a: Double, b: Double, c: Double): Double\n\n/\*\*\n \* Returns the smaller of three values according to the order specified by the given [comparator].\n \* \n \* If there are multiple equal minimal values, returns the first of them.\n \*/\n@SinceKotlin("1.1")\npublic fun <T> minOf(a: T, b: T, c: T, comparator: Comparator<in T>): T {\n return minOf(a, minOf(b, c, comparator), comparator)\n}\n\n/\*\*\n \* Returns the smaller of two values according to the order specified by the given [comparator].\n \* \n \* If values are equal, returns the first one.\n \*/\n@SinceKotlin("1.1")\npublic fun <T> minOf(a: T, b: T, comparator: Comparator<in T>): T {\n return if (comparator.compare(a, b) <= 0) a else b\n}\n\n/\*\*\n \* Returns the smaller of the given values.\n \* \n \* If there are multiple equal minimal values, returns the first of them.\n \*/\n@SinceKotlin("1.4")\npublic expect fun <T : Comparable<T>> minOf(a: T, vararg other: T): T\n\n/\*\*\n \* Returns the smaller of the given values.\n \*/\n@SinceKotlin("1.4")\npublic expect fun minOf(a: Byte, vararg other: Byte): Byte\n\n/\*\*\n \* Returns the smaller of the given values.\n \*/\n@SinceKotlin("1.4")\npublic expect fun minOf(a: Short, vararg other: Short): Short\n\n/\*\*\n \* Returns the smaller of the given values.\n \*/\n@SinceKotlin("1.4")\npublic expect fun minOf(a: Int, vararg other: Int): Int\n\n/\*\*\n \* Returns the smaller of the given values.\n \*/\n@SinceKotlin("1.4")\npublic expect fun minOf(a: Long, vararg other: Long): Long\n\n/\*\*\n \* Returns the smaller of the given values.\n \* \n \* If any value is `NaN`, returns `NaN`.\n \*/\n@SinceKotlin("1.4")\npublic expect fun minOf(a: Float, vararg other: Float): Float\n\n/\*\*\n \* Returns the smaller of the given values.\n \* \n \* If any value is `NaN`, returns `NaN`.\n \*/\n

```

*^@SinceKotlin("1.4")\npublic expect fun minOf(a: Double, vararg other: Double): Double\n\n/**\n * Returns the smaller of the given values according to the order specified by the given [comparator].\n * \n * If there are multiple equal minimal values, returns the first of them.\n */\n@SinceKotlin("1.4")\npublic fun <T> minOf(a: T, vararg other: T, comparator: Comparator<in T>): T {\n    var min = a\n    for (e in other) if (comparator.compare(min, e) > 0) min = e\n    return min\n}\n\n"/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n\n*/\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("MapsKt")\n\npackage kotlin.collections\n\n/\n// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt\n// See: https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\nimport kotlin.random.*\nimport kotlin.ranges.contains\nimport kotlin.ranges.reversed\n\n/**\n * Returns the first non-null value produced by [transform] function being applied to entries of this map in iteration order,\n * or throws [NoSuchElementException] if no non-null value was produced.\n * \n * @sample samples.collections.Collections.Transformations.firstNotNullOf\n */\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun <K, V, R : Any> Map<out K, V>.firstNotNullOf(transform: (Map.Entry<K, V>) -> R?): R {\n    return firstNotNullOfOrNull(transform) ?: throw NoSuchElementException("No element of the map was transformed to a non-null value.")\n}\n\n/**\n * Returns the first non-null value produced by [transform] function being applied to entries of this map in iteration order,\n * or `null` if no non-null value was produced.\n * \n * @sample samples.collections.Collections.Transformations.firstNotNullOf\n */\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun <K, V, R : Any> Map<out K, V>.firstNotNullOfOrNull(transform: (Map.Entry<K, V>) -> R?): R? {\n    for (element in this) {\n        val result = transform(element)\n        if (result != null) {\n            return result\n        }\n    }\n    return null\n}\n\n/**\n * Returns a [List] containing all key-value pairs.\n */\npublic fun <K, V> Map<out K, V>.toList(): List<Pair<K, V>> {\n    if (size == 0)\n        return emptyList()\n    val iterator = entries.iterator()\n    if (!iterator.hasNext())\n        return emptyList()\n    val first = iterator.next()\n    if (!iterator.hasNext())\n        return listOf(first.toPair())\n    val result = ArrayList<Pair<K, V>>(size)\n    result.add(first.toPair())\n    do {\n        result.add(iterator.next().toPair())\n    } while (iterator.hasNext())\n    return result\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each entry of original map.\n * \n * @sample samples.collections.Collections.Transformations.flatMap\n */\npublic inline fun <K, V, R> Map<out K, V>.flatMap(transform: (Map.Entry<K, V>) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each entry of original map.\n * \n * @sample samples.collections.Collections.Transformations.flatMap\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapSequence")\npublic inline fun <K, V, R> Map<out K, V>.flatMap(transform: (Map.Entry<K, V>) -> Sequence<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each entry of original map, to the given [destination].\n */\npublic inline fun <K, V, R, C : MutableCollection<in R>> Map<out K, V>.flatMapTo(destination: C, transform: (Map.Entry<K, V>) -> Iterable<R>): C {\n    for (element in this) {\n        val list = transform(element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each entry of original map, to the given [destination].\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapSequenceTo")\npublic inline fun <K, V, R, C : MutableCollection<in R>> Map<out K, V>.flatMapTo(destination: C, transform: (Map.Entry<K, V>) -> Sequence<R>): C {\n    for (element in this) {\n        val list = transform(element)\n        destination.addAll(list)\n    }\n    return destination\n}\n\n/**\n * Returns a list containing the results of applying the given [transform] function\n * to each entry in the original map.\n * \n * @sample

```

```

samples.collections.Maps.Transformations.mapToList\n */\npublic inline fun <K, V, R> Map<out K,
V>.map(transform: (Map.Entry<K, V>) -> R): List<R> {\n    return mapTo(ArrayList<R>(size),
transform)\n}\n\n/**\n * Returns a list containing only the non-null results of applying the given [transform]
function\n * to each entry in the original map.\n * \n * @sample
samples.collections.Maps.Transformations.mapNotNull\n */\npublic inline fun <K, V, R : Any> Map<out K,
V>.mapNotNull(transform: (Map.Entry<K, V>) -> R?): List<R> {\n    return mapNotNullTo(ArrayList<R>(),
transform)\n}\n\n/**\n * Applies the given [transform] function to each entry in the original map\n * and appends
only the non-null results to the given [destination].\n */\npublic inline fun <K, V, R : Any, C : MutableCollection<in
R>> Map<out K, V>.mapNotNullTo(destination: C, transform: (Map.Entry<K, V>) -> R?): C {\n    forEach {
element -> transform(element)?.let { destination.add(it) } }\n    return destination\n}\n\n/**\n * Applies the given
[transform] function to each entry of the original map\n * and appends the results to the given [destination].\n
*/\npublic inline fun <K, V, R, C : MutableCollection<in R>> Map<out K, V>.mapTo(destination: C, transform:
(Map.Entry<K, V>) -> R): C {\n    for (item in this)\n        destination.add(transform(item))\n    return
destination\n}\n\n/**\n * Returns `true` if all entries match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.all\n */\npublic inline fun <K, V> Map<out K, V>.all(predicate:
(Map.Entry<K, V>) -> Boolean): Boolean {\n    if (isEmpty()) return true\n    for (element in this) if
(!predicate(element)) return false\n    return true\n}\n\n/**\n * Returns `true` if map has at least one entry.\n * \n *
@sample samples.collections.Collections.Aggregates.any\n */\npublic fun <K, V> Map<out K, V>.any(): Boolean
{\n    return !isEmpty()\n}\n\n/**\n * Returns `true` if at least one entry matches the given [predicate].\n * \n *
@sample samples.collections.Collections.Aggregates.anyWithPredicate\n */\npublic inline fun <K, V> Map<out K,
V>.any(predicate: (Map.Entry<K, V>) -> Boolean): Boolean {\n    if (isEmpty()) return false\n    for (element in
this) if (predicate(element)) return true\n    return false\n}\n\n/**\n * Returns the number of entries in this map.\n
*/\n@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K, V>.count(): Int {\n    return size\n}\n\n/**\n
 * Returns the number of entries matching the given [predicate].\n */\npublic inline fun <K, V> Map<out K,
V>.count(predicate: (Map.Entry<K, V>) -> Boolean): Int {\n    if (isEmpty()) return 0\n    var count = 0\n    for
(element in this) if (predicate(element)) ++count\n    return count\n}\n\n/**\n * Performs the given [action] on each
entry.\n */\n@kotlin.internal.HidesMembers\npublic inline fun <K, V> Map<out K, V>.forEach(action:
(Map.Entry<K, V>) -> Unit): Unit {\n    for (element in this) action(element)\n}\n\n@Deprecated("Use
maxByOrNull instead.")\n@DeprecatedSinceKotlin(warningSince =
"1.4", errorSince = "1.5", hiddenSince = "1.6")\n@kotlin.internal.InlineOnly\npublic inline fun <K, V, R :
Comparable<R>> Map<out K, V>.maxBy(selector: (Map.Entry<K, V>) -> R): Map.Entry<K, V>? {\n    return
maxByOrNull(selector)\n}\n\n/**\n * Returns the first entry yielding the largest value of the given function or `null`
if there are no entries.\n * \n * @sample samples.collections.Collections.Aggregates.maxByOrNull\n
*/\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <K, V, R : Comparable<R>> Map<out
K, V>.maxByOrNull(selector: (Map.Entry<K, V>) -> R): Map.Entry<K, V>? {\n    return
entries.maxByOrNull(selector)\n}\n\n/**\n * Returns the largest value among all values produced by [selector]
function\n * applied to each entry in the map.\n * \n * If any of values produced by [selector] function is `NaN`, the
returned result is `NaN`.\n * \n * @throws NoSuchElementException if the map is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K, V>.maxOf(selector:
(Map.Entry<K, V>) -> Double): Double {\n    return entries.maxOf(selector)\n}\n\n/**\n * Returns the largest value
among all values produced by [selector] function\n * applied to each entry in the map.\n * \n * If any of values
produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException
if the map is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K, V>.maxOf(selector:
(Map.Entry<K, V>) -> Float): Float {\n    return entries.maxOf(selector)\n}\n\n/**\n * Returns the largest value
among all values produced by [selector] function\n * applied to each entry in the map.\n * \n * @throws

```

NoSuchElementException if the map is empty.\n

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <K, V, R : Comparable<R>> Map<out K,\nV>.maxOf(selector: (Map.Entry<K, V>) -> R): R {\n    return entries.maxOf(selector)\n}\n\n/**\n * Returns the\nlargest value among all values produced by [selector] function\n * applied to each entry in the map or `null` if there\nare no entries.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K,\nV>.maxOfOrNull(selector: (Map.Entry<K, V>) -> Double): Double? {\n    return\nentries.maxOfOrNull(selector)\n}\n\n/**\n * Returns the largest value among all values produced by [selector]\nfunction\n * applied to each entry in the map or `null` if there are no entries.\n * \n * If any of values produced by\n[selector] function is `NaN`, the returned result is `NaN`.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K,\nV>.maxOfOrNull(selector: (Map.Entry<K, V>) -> Float): Float? {\n    return\nentries.maxOfOrNull(selector)\n}\n\n/**\n * Returns the largest value among all values produced by [selector]\nfunction\n * applied to each entry in the map or `null` if there are no entries.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <K, V, R : Comparable<R>> Map<out K,\nV>.maxOfOrNull(selector: (Map.Entry<K, V>) -> R): R? {\n    return entries.maxOfOrNull(selector)\n}\n\n/**\n * Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]\nfunction applied to each entry in the map.\n * \n * @throws NoSuchElementException if the map is empty.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <K, V, R> Map<out K,\nV>.maxOfWith(comparator: Comparator<in R>, selector: (Map.Entry<K, V>) -> R): R {\n    return\nentries.maxOfWith(comparator, selector)\n}\n\n/**\n * Returns the largest value according to the provided\n[comparator]\n * among all values produced by [selector] function applied to each entry in the map or `null` if there\nare no entries.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <K, V, R> Map<out K,\nV>.maxOfWithOrNull(comparator: Comparator<in R>, selector: (Map.Entry<K, V>) -> R): R? {\n    return\nentries.maxOfWithOrNull(comparator, selector)\n}\n\n@Deprecated("Use maxWithOrNull instead.",\nReplaceWith("this.maxWithOrNull(comparator)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince\n= "1.5", hiddenSince = "1.6")\n@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K,\nV>.maxWith(comparator: Comparator<in Map.Entry<K, V>>): Map.Entry<K, V>? {\n    return\nmaxWithOrNull(comparator)\n}\n\n/**\n * Returns the first entry having the largest value according to the provided\n[comparator] or `null` if there are no entries.\n
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K, V>.maxWithOrNull(comparator: Comparator<in Map.Entry<K, V>>):\nMap.Entry<K, V>? {\n    return entries.maxWithOrNull(comparator)\n}\n\n@Deprecated("Use minByOrNull\ninstead.", ReplaceWith("this.minByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4",\nerrorSince = "1.5", hiddenSince = "1.6")\npublic inline fun <K, V, R : Comparable<R>> Map<out K,\nV>.minBy(selector: (Map.Entry<K, V>) -> R): Map.Entry<K, V>? {\n    return minByOrNull(selector)\n}\n\n/**\n * Returns the first entry yielding the smallest value of the given function or `null` if there are no entries.\n * \n * \n * @sample samples.collections.Collections.Aggregates.minByOrNull\n
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun <K, V, R : Comparable<R>> Map<out\nK, V>.minByOrNull(selector: (Map.Entry<K, V>) -> R): Map.Entry<K, V>? {\n    return\nentries.minByOrNull(selector)\n}\n\n/**\n * Returns the smallest value among all values produced by [selector]\nfunction\n * applied to each entry in the map.\n * \n * If any of values produced by [selector] function is `NaN`, the\n
```

returned result is `NaN`.  
 \* Throws NoSuchElementException if the map is empty.

```

@SinceKotlin("1.4")@OptIn(kotlin.experimental.ExperimentalTypeInference::class)@OverloadResolution
ByLambdaReturnType@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K, V>.minOf(selector:
(Map.Entry<K, V>) -> Double): Double {
    return entries.minOf(selector)
}

```

\* Returns the smallest value among all values produced by [selector] function applied to each entry in the map.  
 \* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.  
 \* Throws NoSuchElementException if the map is empty.

```

@SinceKotlin("1.4")@OptIn(kotlin.experimental.ExperimentalTypeInference::class)@OverloadResolution
ByLambdaReturnType@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K, V>.minOf(selector:
(Map.Entry<K, V>) -> Float): Float {
    return entries.minOf(selector)
}

```

\* Returns the smallest value among all values produced by [selector] function applied to each entry in the map.  
 \* Throws NoSuchElementException if the map is empty.

```

@SinceKotlin("1.4")@OptIn(kotlin.experimental.ExperimentalTypeInference::class)@OverloadResolution
ByLambdaReturnType@kotlin.internal.InlineOnly\npublic inline fun <K, V, R : Comparable<R>> Map<out K,
V>.minOf(selector: (Map.Entry<K, V>) -> R): R {
    return entries.minOf(selector)
}

```

\* Returns the smallest value among all values produced by [selector] function applied to each entry in the map or `null` if there are no entries.  
 \* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.

```

@SinceKotlin("1.4")@OptIn(kotlin.experimental.ExperimentalTypeInference::class)@OverloadResolution
ByLambdaReturnType@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K,
V>.minOrNull(selector: (Map.Entry<K, V>) -> Double): Double? {
    return
entries.minOrNull(selector)
}

```

\* Returns the smallest value among all values produced by [selector] function applied to each entry in the map or `null` if there are no entries.  
 \* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.

```

@SinceKotlin("1.4")@OptIn(kotlin.experimental.ExperimentalTypeInference::class)@OverloadResolution
ByLambdaReturnType@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K,
V>.minOrNull(selector: (Map.Entry<K, V>) -> Float): Float? {
    return
entries.minOrNull(selector)
}

```

\* Returns the smallest value among all values produced by [selector] function applied to each entry in the map or `null` if there are no entries.

```

@SinceKotlin("1.4")@OptIn(kotlin.experimental.ExperimentalTypeInference::class)@OverloadResolution
ByLambdaReturnType@kotlin.internal.InlineOnly\npublic inline fun <K, V, R : Comparable<R>> Map<out K,
V>.minOrNull(selector: (Map.Entry<K, V>) -> R): R? {
    return entries.minOrNull(selector)
}

```

\* Returns the smallest value according to the provided [comparator] among all values produced by [selector] function applied to each entry in the map.  
 \* Throws NoSuchElementException if the map is empty.

```

@SinceKotlin("1.4")@OptIn(kotlin.experimental.ExperimentalTypeInference::class)@OverloadResolution
ByLambdaReturnType@kotlin.internal.InlineOnly\npublic inline fun <K, V, R> Map<out K,
V>.minWith(comparator: Comparator<in R>, selector: (Map.Entry<K, V>) -> R): R {
    return
entries.minWith(comparator, selector)
}

```

\* Returns the smallest value according to the provided [comparator] among all values produced by [selector] function applied to each entry in the map or `null` if there are no entries.

```

@SinceKotlin("1.4")@OptIn(kotlin.experimental.ExperimentalTypeInference::class)@OverloadResolution
ByLambdaReturnType@kotlin.internal.InlineOnly\npublic inline fun <K, V, R> Map<out K,
V>.minWithOrNull(comparator: Comparator<in R>, selector: (Map.Entry<K, V>) -> R): R? {
    return
entries.minWithOrNull(comparator, selector)
}

```

\* Returns the smallest value according to the provided [comparator] among all values produced by [selector] function applied to each entry in the map or `null` if there are no entries.  
 \* Replaced with `this.minWithOrNull(comparator)`.  
 \* @Deprecated("Use minWithOrNull instead.", ReplaceWith("this.minWithOrNull(comparator)"))  
 \* @DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")  
 \* public fun <K, V> Map<out K, V>.minWith(comparator: Comparator<in Map.Entry<K, V>>): Map.Entry<K, V>? {  
 \* return minWithOrNull(comparator)  
 \* }

```

@SinceKotlin("1.4")@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K,

```

```

V>.minWithOrNull(comparator: Comparator<in Map.Entry<K, V>>): Map.Entry<K, V>? {\n  return
entries.minWithOrNull(comparator)\n}\n\n/**\n * Returns `true` if the map has no entries.\n * \n * @sample
samples.collections.Collections.Aggregates.none\n */\npublic fun <K, V> Map<out K, V>.none(): Boolean {\n
return isEmpty()\n}\n\n/**\n * Returns `true` if no entries match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n */\npublic inline fun <K, V> Map<out K,
V>.none(predicate: (Map.Entry<K, V>) -> Boolean): Boolean {\n  if (isEmpty()) return true\n  for (element in
this) if (predicate(element)) return false\n  return true\n}\n\n/**\n * Performs the given [action] on each entry and
returns the map itself afterwards.\n * \n */\n@SinceKotlin("1.1")\npublic inline fun <K, V, M : Map<out K, V>>
M.onEach(action: (Map.Entry<K, V>) -> Unit): M {\n  return apply { for (element in this) action(element)
}\n}\n\n/**\n * Performs the given [action] on each entry, providing sequential index with the entry,\n * and returns
the map itself afterwards.\n * @param [action] function that takes the index of an entry and the entry itself\n * and
performs the action on the entry.\n * \n */\n@SinceKotlin("1.4")\npublic inline fun <K, V, M : Map<out K, V>>
M.onEachIndexed(action: (index: Int, Map.Entry<K, V>) -> Unit): M {\n  return apply {
entries.forEachIndexed(action) }\n}\n\n/**\n * Creates an [Iterable] instance that wraps the original map returning
its entries when being iterated.\n * \n */\n@kotlin.internal.InlineOnly\npublic inline fun <K, V> Map<out K,
V>.asIterable(): Iterable<Map.Entry<K, V>> {\n  return entries\n}\n\n/**\n * Creates a [Sequence] instance that
wraps the original map returning its entries when being iterated.\n * \n */\npublic fun <K, V> Map<out K,
V>.asSequence(): Sequence<Map.Entry<K, V>> {\n  return entries.asSequence()\n}\n\n"/*\n * Copyright 2010-
2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n * \n */\npackage kotlin.text\n\n/\n// NOTE:
THIS FILE IS AUTO-GENERATED by the GenerateUnicodeData.kt\n// See:
https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\n/\n// 10 mappings totally\n\ninternal fun
Char.titlecaseImpl(): String {\n  val uppercase = uppercase()\n  if (uppercase.length > 1) {\n    return if (this ==
"\u0149") uppercase else uppercase[0] + uppercase.substring(1).lowercase()\n  }\n  return
titlecaseChar().toString()\n}\n\n"/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n * \n */\npackage kotlin.text\n\n/**\n * Converts this character to lower case using Unicode
mapping rules of the invariant locale.\n * \n */\n@Deprecated("Use lowercaseChar() instead.")\nReplaceWith("lowercaseChar()")\n@DeprecatedSinceKotlin(warningSince =
"1.5")\n@kotlin.internal.InlineOnly\npublic actual inline fun Char.toLowerCase(): Char =
lowercaseChar()\n\n/**\n * Converts this character to lower case using Unicode mapping rules of the invariant
locale.\n * \n * This function performs one-to-one character mapping.\n * To support one-to-many character
mapping use the [lowercase] function.\n * If this character has no mapping equivalent, the character itself is
returned.\n * \n * @sample samples.text.Chars.lowercase\n
*/\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
actual inline fun Char.lowercaseChar(): Char = lowercase()[0]\n\n/**\n * Converts this character to lower case
using Unicode mapping rules of the invariant locale.\n * \n * This function supports one-to-many character mapping,
thus the length of the returned string can be greater than one.\n * For example, ``\u0130'.lowercase()` returns
``\u0069\u0307``,\n * where ``\u0130`` is the LATIN CAPITAL LETTER I WITH DOT ABOVE character
(`\u0130`).\n * If this character has no lower case mapping, the result of `toString()` of this char is returned.\n
*\n * @sample samples.text.Chars.lowercase\n
*/\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
actual inline fun Char.lowercase(): String = toString().asDynamic().toLowerCase().unsafeCast<String>()\n\n/**\n
* Converts this character to upper case using Unicode mapping rules of the invariant locale.\n
*/\n@Deprecated("Use uppercaseChar() instead.")\nReplaceWith("uppercaseChar()")\n@DeprecatedSinceKotlin(warningSince =
"1.5")\n@kotlin.internal.InlineOnly\npublic actual inline fun Char.toUpperCase(): Char =
uppercaseChar()\n\n/**\n * Converts this character to upper case using Unicode mapping rules of the invariant

```

locale.\n \* This function performs one-to-one character mapping.\n \* To support one-to-many character mapping use the [uppercase] function.\n \* If this character has no mapping equivalent, the character itself is returned.\n \* @sample samples.text.Chars.uppercase\n

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic actual fun Char.uppercaseChar(): Char {\n    val uppercase = uppercase()\n    return if (uppercase.length > 1) this else uppercase[0]\n}\n\n/**\n * Converts this character to upper case using Unicode mapping rules of the invariant locale.\n * This function supports one-to-many character mapping, thus the length of the returned string can be greater than one.\n * For example, `'\u0046\u0046'.uppercase()` returns `'\u0046\u0046'`,\n * where `'\u0046\u0046'` is the LATIN SMALL LIGATURE FF character (`\u0046`).\n * If this character has no upper case mapping, the result of `toString()` of this char is returned.\n * @sample samples.text.Chars.uppercase\n
```

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic actual inline fun Char.uppercase(): String = toString().asDynamic().toUpperCase().unsafeCast<String>()\n\n/**\n * Converts this character to title case using Unicode mapping rules of the invariant locale.\n * This function performs one-to-one character mapping.\n * To support one-to-many character mapping use the [titlecase] function.\n * If this character has no mapping equivalent, the result of calling [uppercaseChar] is returned.\n * @sample samples.text.Chars.titlecase\n
```

```
*\n@SinceKotlin("1.5")\npublic actual fun Char.titlecaseChar(): Char = titlecaseCharImpl()\n\n/**\n * Returns `true` if this character is a Unicode high-surrogate code unit (also known as leading-surrogate code unit).\n */\npublic actual fun Char.isHighSurrogate(): Boolean = this in Char.MIN_HIGH_SURROGATE..Char.MAX_HIGH_SURROGATE\n\n/**\n * Returns `true` if this character is a Unicode low-surrogate code unit (also known as trailing-surrogate code unit).\n */\npublic actual fun Char.isLowSurrogate(): Boolean = this in Char.MIN_LOW_SURROGATE..Char.MAX_LOW_SURROGATE\n\n/**\n * Returns the Unicode general category of this character.\n */\n@SinceKotlin("1.5")\npublic actual val Char.category: CharCategory\n    get() = CharCategory.valueOf(getCategoryValue())\n\n/**\n * Returns `true` if this character (Unicode code point) is defined in Unicode.\n * A character is considered to be defined in Unicode if its [category] is not [CharCategory.UNASSIGNED].\n */\n@SinceKotlin("1.5")\npublic actual fun Char.isDefined(): Boolean {\n    if (this < '\u0080') {\n        return true\n    }\n    return getCategoryValue() != CharCategory.UNASSIGNED.value\n}\n\n/**\n * Returns `true` if this character is a letter.\n * A character is considered to be a letter if its [category] is [CharCategory.UPPERCASE_LETTER], [CharCategory.LOWERCASE_LETTER], [CharCategory.TITLECASE_LETTER], [CharCategory.MODIFIER_LETTER], or [CharCategory.OTHER_LETTER].\n * @sample samples.text.Chars.isLetter\n
```

```
*\n@SinceKotlin("1.5")\npublic actual fun Char.isLetter(): Boolean {\n    if (this in 'a'..'z' || this in 'A'..'Z') {\n        return true\n    }\n    if (this < '\u0080') {\n        return false\n    }\n    return isLetterImpl()\n}\n\n/**\n * Returns `true` if this character is a letter or digit.\n * @see isLetter\n * @see isDigit\n
```

```
*\n@sample samples.text.Chars.isLetterOrDigit\n
```

```
*\n@SinceKotlin("1.5")\npublic actual fun Char.isLetterOrDigit(): Boolean {\n    if (this in 'a'..'z' || this in 'A'..'Z' || this in '0'..'9') {\n        return true\n    }\n    if (this < '\u0080') {\n        return false\n    }\n    return isDigitImpl() || isLetterImpl()\n}\n\n/**\n * Returns `true` if this character is a digit.\n * A character is considered to be a digit if its [category] is [CharCategory.DECIMAL_DIGIT_NUMBER].\n * @sample samples.text.Chars.isDigit\n
```

```
*\n@SinceKotlin("1.5")\npublic actual fun Char.isDigit(): Boolean {\n    if (this in '0'..'9') {\n        return true\n    }\n    if (this < '\u0080') {\n        return false\n    }\n    return isDigitImpl()\n}\n\n/**\n * Returns `true` if this character is upper case.\n * A character is considered to be an upper case character if its [category] is [CharCategory.UPPERCASE_LETTER],\n * or it has contributory property Other_Uppercase as defined by the Unicode Standard.\n * @sample samples.text.Chars.isUpperCase\n
```

```
*\n@SinceKotlin("1.5")\npublic actual fun Char.isUpperCase(): Boolean {\n    if (this in 'A'..'Z') {\n        return true\n    }\n    if (this < '\u0080') {\n        return false\n    }\n    return isUpperCaseImpl()\n}\n\n/**\n * Returns `true` if this character is lower case.\n * A character is considered to be a lower case character if its [category] is [CharCategory.LOWERCASE_LETTER],\n * or it has contributory property Other_Lowercase as defined by the Unicode Standard.\n * @sample
```

```

samples.text.Chars.isLowerCase\n *\n@SinceKotlin("1.5")\npublic actual fun Char.isLowerCase(): Boolean {\n
if (this in 'a'..'z') {\n    return true\n } \n if (this < "\u0080") {\n    return false\n }\n return
isLowerCaseImpl()\n}\n\n/**\n * Returns `true` if this character is a title case letter.\n *\n * A character is
considered to be a title case letter if its [category] is [CharCategory.TITLECASE_LETTER].\n *\n * @sample
samples.text.Chars.isTitleCase\n *\n@SinceKotlin("1.5")\npublic actual fun Char.isTitleCase(): Boolean {\n
if (this < "\u0080") {\n    return false\n }\n return getCategoryValue() ==
CharCategory.TITLECASE_LETTER.value\n}\n\n/**\n * Returns `true` if this character is an ISO control
character.\n *\n * A character is considered to be an ISO control character if its [category] is
[CharCategory.CONTROL],\n *\n meaning the Char is in the range "\u0000'..\u001F'" or in the range
"\u007F'..\u009F".\n *\n * @sample samples.text.Chars.isISOControl\n *\n@SinceKotlin("1.5")\npublic actual
fun Char.isISOControl(): Boolean {\n    return this <= "\u001F" || this in "\u007F'..\u009F"\n}\n\n/**\n *
Determines whether a character is whitespace according to the Unicode standard.\n *\n Returns `true` if the character
is whitespace.\n *\n * @sample samples.text.Chars.isWhitespace\n *\npublic actual fun Char.isWhitespace():
Boolean = isWhitespaceImpl()\n\n/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n *\n Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n *\n\npackage kotlin.text\n\nimport kotlin.js.RegExp\n\n/**\n * Converts the characters
in the specified array to a string.\n *\n@SinceKotlin("1.2")\n@Deprecated("Use CharArray.concatToString()
instead", ReplaceWith("chars.concatToString()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince
= "1.5")\npublic actual fun String(chars: CharArray): String {\n    var result = ""\n    for (char in chars) {\n
result += char\n    }\n    return result\n}\n\n/**\n * Converts the characters from a portion of the specified array to a
string.\n *\n * @throws IndexOutOfBoundsException if either [offset] or [length] are less than zero\n * or `offset +
length` is out of [chars] array bounds.\n *\n@SinceKotlin("1.2")\n@Deprecated("Use
CharArray.concatToString(startIndex, endIndex) instead", ReplaceWith("chars.concatToString(offset, offset +
length)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5")\npublic actual fun String(chars:
CharArray, offset: Int, length: Int): String {\n    if (offset < 0 || length < 0 || chars.size - offset < length)\n
throw IndexOutOfBoundsException("size: ${chars.size}; offset: $offset; length: $length")\n    var result = ""\n
for (index in offset until offset + length) {\n        result += chars[index]\n    }\n    return result\n}\n\n/**\n *
Concatenates characters in this [CharArray] into a String.\n\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic actual fun
CharArray.concatToString(): String {\n    var result = ""\n    for (char in this) {\n        result += char\n    }\n
return result\n}\n\n/**\n * Concatenates characters in this [CharArray] or its subrange into a String.\n *\n * @param
startIndex the beginning (inclusive) of the subrange of characters, 0 by default.\n * @param endIndex the end
(exclusive) of the subrange of characters, size of this array by default.\n *\n * @throws
IndexOutOfBoundsException if [startIndex] is less than zero or [endIndex] is greater than the size of this array.\n *\n
* @throws IllegalArgumentException if [startIndex] is greater than [endIndex].\n\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic actual fun CharArray.concatToString(startIndex: Int = 0,
endIndex: Int = this.size): String {\n    AbstractList.checkBoundsIndexes(startIndex, endIndex, this.size)\n    var
result = ""\n    for (index in startIndex until endIndex) {\n        result += this[index]\n    }\n    return
result\n}\n\n/**\n * Returns a [CharArray] containing characters of this string.\n\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic actual fun
String.toCharArray(): CharArray {\n    return CharArray(length) { get(it) }\n}\n\n/**\n * Returns a [CharArray]
containing characters of this string or its substring.\n *\n * @param startIndex the beginning (inclusive) of the
substring, 0 by default.\n * @param endIndex the end (exclusive) of the substring, length of this string by default.\n
*\n * @throws IndexOutOfBoundsException if [startIndex] is less than zero or [endIndex] is greater than the length
of this string.\n *\n * @throws IllegalArgumentException if [startIndex] is greater than [endIndex].\n\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic actual fun String.toCharArray(startIndex: Int = 0, endIndex: Int

```

```
= this.length): CharArray {\n  AbstractList.checkBoundsIndexes(startIndex, endIndex, length)\n  return CharArray(endIndex - startIndex) { get(startIndex + it) }\n}\n\n/**\n * Decodes a string from the bytes in UTF-8 encoding in this array.\n *\n * Malformed byte sequences are replaced by the replacement char `\\uFFFD`.\n */\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic actual fun ByteArray.decodeToString(): String {\n  return decodeUtf8(this, 0, size, false)\n}\n\n/**\n * Decodes a string from the bytes in UTF-8 encoding in this array or its subrange.\n *\n * @param startIndex the beginning (inclusive) of the subrange to decode, 0 by default.\n * @param endIndex the end (exclusive) of the subrange to decode, size of this array by default.\n * @param throwOnInvalidSequence specifies whether to throw an exception on malformed byte sequence or replace it by the replacement char `\\uFFFD`.\n *\n * @throws IndexOutOfBoundsException if [startIndex] is less than zero or [endIndex] is greater than the size of this array.\n * @throws IllegalArgumentException if [startIndex] is greater than [endIndex].\n * @throws CharacterCodingException if the byte array contains malformed UTF-8 byte sequence and [throwOnInvalidSequence] is true.\n */\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic actual fun ByteArray.decodeToString(\n  startIndex: Int = 0,\n  endIndex: Int = this.size,\n  throwOnInvalidSequence: Boolean = false\n): String {\n  AbstractList.checkBoundsIndexes(startIndex, endIndex, this.size)\n  return decodeUtf8(this, startIndex, endIndex, throwOnInvalidSequence)\n}\n\n/**\n * Encodes this string to an array of bytes in UTF-8 encoding.\n *\n * Any malformed char sequence is replaced by the replacement byte sequence.\n */\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic actual fun String.encodeToByteArray(): ByteArray {\n  return encodeUtf8(this, 0, length, false)\n}\n\n/**\n * Encodes this string or its substring to an array of bytes in UTF-8 encoding.\n *\n * @param startIndex the beginning (inclusive) of the substring to encode, 0 by default.\n * @param endIndex the end (exclusive) of the substring to encode, length of this string by default.\n * @param throwOnInvalidSequence specifies whether to throw an exception on malformed char sequence or replace.\n *\n * @throws IndexOutOfBoundsException if [startIndex] is less than zero or [endIndex] is greater than the length of this string.\n * @throws IllegalArgumentException if [startIndex] is greater than [endIndex].\n * @throws CharacterCodingException if this string contains malformed char sequence and [throwOnInvalidSequence] is true.\n */\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic actual fun String.encodeToByteArray(\n  startIndex: Int = 0,\n  endIndex: Int = this.length,\n  throwOnInvalidSequence: Boolean = false\n): ByteArray {\n  AbstractList.checkBoundsIndexes(startIndex, endIndex, length)\n  return encodeUtf8(this, startIndex, endIndex, throwOnInvalidSequence)\n}\n\n/**\n * Returns a copy of this string converted to upper case using the rules of the default locale.\n */\n@Deprecated("Use uppercase() instead.")\nReplaceWith("uppercase()")\n@DeprecatedSinceKotlin(warningSince = "1.5")\n@kotlin.internal.InlineOnly\npublic actual inline fun String.toUpperCase(): String = asDynamic().toUpperCase()\n\n/**\n * Returns a copy of this string converted to upper case using Unicode mapping rules of the invariant locale.\n *\n * This function supports one-to-many and many-to-one character mapping,\n * thus the length of the returned string can be different from the length of the original string.\n *\n * @sample samples.text.Strings.uppercase\n */\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic actual inline fun String.uppercase(): String = asDynamic().toUpperCase()\n\n/**\n * Returns a copy of this string converted to lower case using the rules of the default locale.\n */\n@Deprecated("Use lowercase() instead.")\nReplaceWith("lowercase()")\n@DeprecatedSinceKotlin(warningSince = "1.5")\n@kotlin.internal.InlineOnly\npublic actual inline fun String.toLowerCase(): String = asDynamic().toLowerCase()\n\n/**\n * Returns a copy of this string converted to lower case using Unicode mapping rules of the invariant locale.\n *\n * This function supports one-to-many and many-to-one character mapping,\n * thus the length of the returned string can be different from the length of the original string.\n *\n * @sample samples.text.Strings.lowercase\n */\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic actual inline fun String.toLowerCase(): String = asDynamic().toLowerCase()\n
```

```

*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
actual inline fun String.lowercase(): String = asDynamic().toLowerCase()\n\n@kotlin.internal.InlineOnly\ninternal
actual inline fun String.nativeIndexOf(str: String, fromIndex: Int): Int = asDynamic().indexOf(str,
fromIndex)\n\n@kotlin.internal.InlineOnly\ninternal actual inline fun String.nativeLastIndexOf(str: String,
fromIndex: Int): Int = asDynamic().lastIndexOf(str, fromIndex)\n\n@kotlin.internal.InlineOnly\ninternal inline fun
String.nativeStartsWith(s: String, position: Int): Boolean = asDynamic().startsWith(s,
position)\n\n@kotlin.internal.InlineOnly\ninternal inline fun String.nativeEndsWith(s: String): Boolean =
asDynamic().endsWith(s)\n\n@kotlin.internal.InlineOnly\npublic actual inline fun String.substring(startIndex: Int):
String = asDynamic().substring(startIndex)\n\n@kotlin.internal.InlineOnly\npublic actual inline fun
String.substring(startIndex: Int, endIndex: Int): String = asDynamic().substring(startIndex,
endIndex)\n\n@Deprecated("Use String.plus() instead", ReplaceWith("this +
str"))\n@DeprecatedSinceKotlin(warningSince = "1.6")\n@kotlin.internal.InlineOnly\npublic inline fun
String.concat(str: String): String = asDynamic().concat(str)\n\n@Deprecated("Use Regex.findAll() instead or
invoke matches() on String dynamically:
this.asDynamic().match(regex)")\n@DeprecatedSinceKotlin(warningSince =
"1.6")\n@kotlin.internal.InlineOnly\npublic inline fun String.match(regex: String): Array<String>? =
asDynamic().match(regex)\n\n//native public fun String.trim(): String\n//TODO: String.replace to implement
effective trimLeading and trimTrailing\n\n@kotlin.internal.InlineOnly\ninternal inline fun
String.nativeReplace(pattern: RegExp, replacement: String): String = asDynamic().replace(pattern,
replacement)\n\n/**\n * Compares two strings lexicographically, optionally ignoring case differences.\n * If
[ignoreCase] is true, the result of `Char.toUpperCaseChar().toLowerCaseChar()` on each character is compared.\n
*\n@SinceKotlin("1.2")\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\npublic
actual fun String.compareTo(other: String, ignoreCase: Boolean = false): Int {\n    if (ignoreCase) {\n        val n1 =
this.length\n        val n2 = other.length\n        val min = minOf(n1, n2)\n        if (min == 0) return n1 - n2\n        for
(index in 0 until min) {\n            var thisChar = this[index]\n            var otherChar = other[index]\n            if
(thisChar != otherChar) {\n                thisChar = thisChar.toUpperCaseChar()\n                otherChar =
otherChar.toUpperCaseChar()\n                if (thisChar != otherChar) {\n                    thisChar =
thisChar.toLowerCaseChar()\n                    otherChar = otherChar.toLowerCaseChar()\n                    if (thisChar !=
otherChar) {\n                        return thisChar.compareTo(otherChar)\n                    }\n                }\n            }
\n        } else {\n            return compareTo(other)\n        }\n    }\n}\n\n/**\n * Returns `true` if the contents
of this char sequence are equal to the contents of the specified [other],\n * i.e. both char sequences contain the same
number of the same characters in the same order.\n * @sample samples.text.Strings.contentEquals\n
*\n@SinceKotlin("1.5")\npublic actual infix fun CharSequence?.contentEquals(other: CharSequence?): Boolean =
contentEqualsImpl(other)\n\n/**\n * Returns `true` if the contents of this char sequence are equal to the contents of
the specified [other], optionally ignoring case difference.\n * @param ignoreCase `true` to ignore character case
when comparing contents.\n * @sample samples.text.Strings.contentEquals\n
*\n@SinceKotlin("1.5")\npublic
actual fun CharSequence?.contentEquals(other: CharSequence?, ignoreCase: Boolean): Boolean {\n    return if
(ignoreCase)\n        this.contentEqualsIgnoreCaseImpl(other)\n    else\n        this.contentEqualsImpl(other)\n}\n\nprivate val STRING_CASE_INSENSITIVE_ORDER = Comparator<String>
{ a, b -> a.compareTo(b, ignoreCase = true) }\n\n@SinceKotlin("1.2")\npublic actual val
String.Companion.CASE_INSENSITIVE_ORDER: Comparator<String>\n    get() =
STRING_CASE_INSENSITIVE_ORDER\n", "/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n
*\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("CharsKt")\n\npackage kotlin.text\n\n/**\n
 * Returns the numeric value of the decimal digit that this Char represents.\n * Throws an exception if this Char is
not a valid decimal digit.\n * A Char is considered to represent a decimal digit if [isDigit] is true for the Char.\n
 * In this case, the Unicode decimal digit value of the character is returned.\n * @sample

```

samples.text.Chars.digitToInt

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun Char.digitToInt(): Int {\n    return digitOf(this, 10).also { \n        if (it < 0) throw IllegalArgumentException("\nChar $this is not a decimal digit")\n    }\n}\n\n/**\n * Returns the numeric value of the digit that this Char represents in the specified [radix].\n * Throws an exception if the [radix] is not in the range `2..36` or if this Char is not a valid digit in the specified [radix].\n * A Char is considered to represent a digit in the specified [radix] if at least one of the following is true:\n * - [isDigit] is `true` for the Char and the Unicode decimal digit value of the character is less than the specified [radix]. In this case the decimal digit value is returned.\n * - The Char is one of the uppercase Latin letters 'A' through 'Z' and its [code] is less than `radix + 'A'.code - 10`. In this case, `this.code - 'A'.code + 10` is returned.\n * - The Char is one of the lowercase Latin letters 'a' through 'z' and its [code] is less than `radix + 'a'.code - 10`. In this case, `this.code - 'a'.code + 10` is returned.\n * - The Char is one of the fullwidth Latin capital letters '\uFF21' through '\uFF3A' and its [code] is less than `radix + 0xFF21 - 10`. In this case, `this.code - 0xFF21 + 10` is returned.\n * - The Char is one of the fullwidth Latin small letters '\uFF41' through '\uFF5A' and its [code] is less than `radix + 0xFF41 - 10`. In this case, `this.code - 0xFF41 + 10` is returned.\n */\n * @sample
```

samples.text.Chars.digitToInt

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun Char.digitToInt(radix: Int): Int {\n    return digitToIntOrNull(radix) ?: throw IllegalArgumentException("\nChar $this is not a digit in the given radix=$radix")\n}\n\n/**\n * Returns the numeric value of the decimal digit that this Char represents, or `null` if this Char is not a valid decimal digit.\n * A Char is considered to represent a decimal digit if [isDigit] is true for the Char.\n * In this case, the Unicode decimal digit value of the character is returned.\n */\n * @sample
```

samples.text.Chars.digitToIntOrNull

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun Char.digitToIntOrNull(): Int? {\n    return digitOf(this, 10).takeIf { it >= 0 }\n}\n\n/**\n * Returns the numeric value of the digit that this Char represents in the specified [radix], or `null` if this Char is not a valid digit in the specified [radix].\n * Throws an exception if the [radix] is not in the range `2..36`.\n * A Char is considered to represent a digit in the specified [radix] if at least one of the following is true:\n * - [isDigit] is `true` for the Char and the Unicode decimal digit value of the character is less than the specified [radix]. In this case the decimal digit value is returned.\n * - The Char is one of the uppercase Latin letters 'A' through 'Z' and its [code] is less than `radix + 'A'.code - 10`. In this case, `this.code - 'A'.code + 10` is returned.\n * - The Char is one of the lowercase Latin letters 'a' through 'z' and its [code] is less than `radix + 'a'.code - 10`. In this case, `this.code - 'a'.code + 10` is returned.\n * - The Char is one of the fullwidth Latin capital letters '\uFF21' through '\uFF3A' and its [code] is less than `radix + 0xFF21 - 10`. In this case, `this.code - 0xFF21 + 10` is returned.\n * - The Char is one of the fullwidth Latin small letters '\uFF41' through '\uFF5A' and its [code] is less than `radix + 0xFF41 - 10`. In this case, `this.code - 0xFF41 + 10` is returned.\n */\n * @sample
```

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
```

```
Char.digitToIntOrNull(radix: Int): Int? {\n    checkRadix(radix)\n    return digitOf(this, radix).takeIf { it >= 0 }\n}\n\n/**\n * Returns the Char that represents this decimal digit.\n * Throws an exception if this value is not in the range `0..9`.\n * If this value is in `0..9`, the decimal digit Char with code `0'.code + this` is returned.\n */\n * @sample
```

samples.text.Chars.digitToChar

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun Int.digitToChar(): Char {\n    if (this in 0..9) {\n        return '0' + this\n    }\n    throw IllegalArgumentException("\nInt $this is not a decimal digit")\n}\n\n/**\n * Returns the Char that represents this numeric digit value in the specified [radix].\n * Throws an exception if the [radix] is not in the range `2..36` or if this value is not in the range `0 until radix`.\n * If this value is less than `10`, the decimal digit Char with code `0'.code + this` is returned.\n * Otherwise, the uppercase Latin letter with code `A'.code + this - 10` is returned.\n */\n * @sample
```

```
samples.text.Chars.digitToChar
```

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun Int.digitToChar(radix: Int): Char {\n    if (radix !in 2..36) {\n        throw IllegalArgumentException("\nInvalid radix: $radix. Valid radix values are in range 2..36")\n    }\n    if (this < 0 || this >= radix) {\n        throw IllegalArgumentException("\nDigit
```

```

$this does not represent a valid digit in radix $radix`)
}
return if (this < 10) {
    '0' + this
} else {
    'A' + this - 10
}
}
}
}

Char.toLowerCase(): Char
* Converts this character to lower case using Unicode mapping rules of the invariant locale.
* @Deprecated("Use lowercaseChar() instead.")
ReplaceWith("lowercaseChar()")
}
}
}

Char.toLowerCase(): Char
* Converts this character to lower case using Unicode mapping rules of the invariant locale.
* This function performs one-to-one character mapping.
* To support one-to-many character mapping use the [toLowerCase] function.
* If this character has no mapping equivalent, the character itself is returned.
* @sample samples.text.Chars.toLowerCase

*
* @SinceKotlin("1.5")
* @WasExperimental(ExperimentalStdlibApi::class)
}
}
}

Char.toLowerCaseChar(): Char
* Converts this character to lower case using Unicode mapping rules of the invariant locale.
* This function supports one-to-many character mapping, thus the length of the returned string can be greater than one.
* For example, `'\u0130'.toLowerCase()` returns `'\u0069\u0307'`, where `'\u0130` is the LATIN CAPITAL LETTER I WITH DOT ABOVE character (`\ufffd\u0130`).
* If this character has no lower case mapping, the result of `toString()` of this char is returned.
* @sample samples.text.Chars.toLowerCaseChar

*
* @SinceKotlin("1.5")
* @WasExperimental(ExperimentalStdlibApi::class)
}
}
}

Char.lowercase(): String
* Converts this character to upper case using Unicode mapping rules of the invariant locale.
* @Deprecated("Use uppercaseChar() instead.")
ReplaceWith("uppercaseChar()")
}
}
}

Char.toUpperCase(): Char
* Converts this character to upper case using Unicode mapping rules of the invariant locale.
* This function performs one-to-one character mapping.
* To support one-to-many character mapping use the [toUpperCase] function.
* If this character has no mapping equivalent, the character itself is returned.
* @sample samples.text.Chars.toUpperCase

*
* @SinceKotlin("1.5")
* @WasExperimental(ExperimentalStdlibApi::class)
}
}
}

Char.toUpperCaseChar(): Char
* Converts this character to upper case using Unicode mapping rules of the invariant locale.
* This function supports one-to-many character mapping, thus the length of the returned string can be greater than one.
* For example, `'\uFB00'.toUpperCase()` returns `'\u0046\u0046'`, where `'\uFB00` is the LATIN SMALL LIGATURE FF character (`\ufffd\u0046\u0046`).
* If this character has no upper case mapping, the result of `toString()` of this char is returned.
* @sample samples.text.Chars.toUpperCaseChar

*
* @SinceKotlin("1.5")
* @WasExperimental(ExperimentalStdlibApi::class)
}
}
}

Char.toUpperCase(): String
* Converts this character to title case using Unicode mapping rules of the invariant locale.
* This function performs one-to-one character mapping.
* To support one-to-many character mapping use the [toUpperCase] function.
* If this character has no mapping equivalent, the result of calling [toUpperCaseChar] is returned.
* @sample samples.text.Chars.toUpperCase

*
* @SinceKotlin("1.5")
}
}
}

Char.titlecaseChar(): Char
* Converts this character to title case using Unicode mapping rules of the invariant locale.
* This function supports one-to-many character mapping, thus the length of the returned string can be greater than one.
* For example, `'\uFB00'.titlecase()` returns `'\u0046\u0066'`, where `'\uFB00` is the LATIN SMALL LIGATURE FF character (`\ufffd\u0046\u0066`).
* If this character has no title case mapping, the result of [toUpperCase] is returned instead.
* @sample samples.text.Chars.titlecaseChar

*
* @SinceKotlin("1.5")
}
}
}

Char.titlecase(): String = titlecaseImpl()
}
}
}

Char.plus(other: String): String = this.toString() + other
}
}
}

Char.equals(other: Char, ignoreCase: Boolean = false): Boolean
{
    if (this == other) return true
    if (!ignoreCase) return false
    val thisUpper = this.toUpperCaseChar()
    val otherUpper = other.toUpperCaseChar()
    return thisUpper == otherUpper || thisUpper.toLowerCaseChar() ==

```

```

otherUpper.lowercaseChar()\n}\n\n/**\n * Returns `true` if this character is a Unicode surrogate code unit.\n
*\npublic fun Char.isSurrogate(): Boolean = this in Char.MIN_SURROGATE..Char.MAX_SURROGATE\n\n/**\n
* Returns the Unicode general category of this character.\n *\n@SinceKotlin("1.5")\npublic expect val
Char.category: CharCategory\n\n/**\n * Returns `true` if this character (Unicode code point) is defined in
Unicode.\n *\n * A character is considered to be defined in Unicode if its [category] is not
[CharCategory.UNASSIGNED].\n *\n@SinceKotlin("1.5")\npublic expect fun Char.isDefined():
Boolean\n\n/**\n * Returns `true` if this character is a letter.\n *\n * A character is considered to be a letter if its
[category] is [CharCategory.UPPERCASE_LETTER],\n * [CharCategory.LOWERCASE_LETTER],
[CharCategory.TITLECASE_LETTER], [CharCategory.MODIFIER_LETTER], or
[CharCategory.OTHER_LETTER].\n *\n * @sample samples.text.Chars.isLetter\n
*\n@SinceKotlin("1.5")\npublic expect fun Char.isLetter(): Boolean\n\n/**\n * Returns `true` if this character is a
letter or digit.\n *\n * @see isLetter\n * @see isDigit\n *\n * @sample samples.text.Chars.isLetterOrDigit\n
*\n@SinceKotlin("1.5")\npublic expect fun Char.isLetterOrDigit(): Boolean\n\n/**\n * Returns `true` if this
character is a digit.\n *\n * A character is considered to be a digit if its [category] is
[CharCategory.DECIMAL_DIGIT_NUMBER].\n *\n * @sample samples.text.Chars.isDigit\n
*\n@SinceKotlin("1.5")\npublic expect fun Char.isDigit(): Boolean\n\n/**\n * Returns `true` if this character is
upper case.\n *\n * A character is considered to be an upper case character if its [category] is
[CharCategory.UPPERCASE_LETTER],\n * or it has contributory property Other_Uppercase as defined by the
Unicode Standard.\n *\n * @sample samples.text.Chars.isUpperCase\n *\n@SinceKotlin("1.5")\npublic expect
fun Char.isUpperCase(): Boolean\n\n/**\n * Returns `true` if this character is lower case.\n *\n * A character is
considered to be a lower case character if its [category] is [CharCategory.LOWERCASE_LETTER],\n * or it has
contributory property Other_Lowercase as defined by the Unicode Standard.\n *\n * @sample
samples.text.Chars.isLowerCase\n *\n@SinceKotlin("1.5")\npublic expect fun Char.isLowerCase():
Boolean\n\n/**\n * Returns `true` if this character is a title case letter.\n *\n * A character is considered to be a title
case letter if its [category] is [CharCategory.TITLECASE_LETTER].\n *\n * @sample
samples.text.Chars.isTitleCase\n *\n@SinceKotlin("1.5")\npublic expect fun Char.isTitleCase(): Boolean\n\n/**\n
* Returns `true` if this character is an ISO control character.\n *\n * A character is considered to be an ISO control
character if its [category] is [CharCategory.CONTROL],\n * meaning the Char is in the range ``\u0000'..' \u001F``
or in the range ``\u007F'..' \u009F``.\n *\n * @sample samples.text.Chars.isISOControl\n
*\n@SinceKotlin("1.5")\npublic expect fun Char.isISOControl(): Boolean\n\n/**\n * Determines whether a
character is whitespace according to the Unicode standard.\n * Returns `true` if the character is whitespace.\n *\n *
@sample samples.text.Chars.isWhitespace\n *\npublic expect fun Char.isWhitespace(): Boolean\n", "/*\n
* Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is
governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n *\n@npackage
kotlin\n\n/**\n * Creates a Char with the specified [code], or throws an exception if the [code] is out of
`Char.MIN_VALUE.code..Char.MAX_VALUE.code`. \n *\n * If the program that calls this function is written in a
way that only valid [code] is passed as the argument,\n *\n * using the overload that takes a [UShort] argument is
preferable (`Char(intValue.toUShort())`).\n *\n * That overload doesn't check validity of the argument, and may
improve program performance when the function is called routinely inside a loop.\n *\n * @sample
samples.text.Chars.charFromCode\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun Char(code: Int): Char { \n    if (code < Char.MIN_VALUE.code || code > Char.MAX_VALUE.code) {\n
        throw IllegalArgumentException("Invalid Char code: $code")\n    }\n    return code.toChar()\n}\n\n/**\n
* Creates a Char with the specified [code].\n *\n * @sample samples.text.Chars.charFromCode\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\n@Suppress("NO_ACTUAL_FOR
_EXPECT")\npublic expect fun Char(code: UShort): Char\n\n/**\n * Returns the code of this Char.\n *\n * Code of
a Char is the value it was constructed with, and the UTF-16 code unit corresponding to this Char.\n *\n * @sample
samples.text.Chars.code\n

```

```

*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\n@Suppress("DEPRECATION")\npublic inline val Char.code: Int get() = this.toInt()\n", /*\n * Copyright 2010-2021\n * JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the\n * Apache 2.0 license that can be found in the license/LICENSE.txt file.\n *\n *\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("SequencesKt")\n\npackage\nkotlin.sequences\n\n/\n// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt\n// See:\nhttps://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\nimport kotlin.random.*\n\n * Returns\n`true` if [element] is found in the sequence.\n * The operation is _terminal_.\n\n * public operator fun\n<@kotlin.internal.OnlyInputTypes T> Sequence<T>.contains(element: T): Boolean {\n    return indexOf(element)\n    >= 0\n}\n\n * Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the\n[index] is out of bounds of this sequence.\n * The operation is _terminal_.\n * \n * @sample\nsamples.collections.Collections.Elements.elementAt\n\n * public fun <T> Sequence<T>.elementAt(index: Int): T\n{\n    return elementAtOrElse(index) { throw IndexOutOfBoundsException("Sequence doesn't contain element at\nindex $index.") }\n}\n\n * Returns an element at the given [index] or the result of calling the [defaultValue]\nfunction if the [index] is out of bounds of this sequence.\n * The operation is _terminal_.\n * \n * @sample\nsamples.collections.Collections.Elements.elementAtOrElse\n\n * public fun <T>\nSequence<T>.elementAtOrElse(index: Int, defaultValue: (Int) -> T): T {\n    if (index < 0)\n        return\n        defaultValue(index)\n    val iterator = iterator()\n    var count = 0\n    while (iterator.hasNext()) {\n        val element\n        = iterator.next()\n        if (index == count++)\n            return element\n    }\n    return\n    defaultValue(index)\n}\n\n * Returns an element at the given [index] or `null` if the [index] is out of bounds of\nthis sequence.\n * The operation is _terminal_.\n * \n * @sample\nsamples.collections.Collections.Elements.elementAtOrNull\n\n * public fun <T>\nSequence<T>.elementAtOrNull(index: Int): T? {\n    if (index < 0)\n        return null\n    val iterator = iterator()\n    var count = 0\n    while (iterator.hasNext()) {\n        val element = iterator.next()\n        if (index == count++)\n            return element\n    }\n    return null\n}\n\n * Returns the first element matching the given [predicate], or `null`\nif no such element was found.\n * The operation is _terminal_.\n * \n * @sample\nsamples.collections.Collections.Elements.find\n\n * @kotlin.internal.InlineOnly\npublic inline fun <T>\nSequence<T>.find(predicate: (T) -> Boolean): T? {\n    return firstOrNull(predicate)\n}\n\n * Returns the last\n * element matching the given [predicate], or `null` if no such element was found.\n * The operation is\n * _terminal_.\n * \n * @sample\nsamples.collections.Collections.Elements.find\n\n * @kotlin.internal.InlineOnly\npublic inline fun <T> Sequence<T>.findLast(predicate: (T) -> Boolean): T? {\n    return lastOrNull(predicate)\n}\n\n * Returns first element.\n * @throws [NoSuchElementException] if the\n * sequence is empty.\n * The operation is _terminal_.\n\n * public fun <T> Sequence<T>.first(): T {\n    val\n    iterator = iterator()\n    if (!iterator.hasNext())\n        throw NoSuchElementException("Sequence is empty.")\n    return iterator.next()\n}\n\n * Returns the first element matching the given [predicate].\n * @throws\n * [NoSuchElementException] if no such element is found.\n * The operation is _terminal_.\n\n * public inline fun\n * <T> Sequence<T>.first(predicate: (T) -> Boolean): T {\n    for (element in this) if (predicate(element)) return\n    element\n    throw NoSuchElementException("Sequence contains no element matching the predicate.")\n}\n\n * Returns the first non-null value produced by [transform] function being applied to elements of this sequence in\n * iteration order,\n * or throws [NoSuchElementException] if no non-null value was produced.\n * The operation\n * is _terminal_.\n * \n * @sample\nsamples.collections.Collections.Transformations.firstNotNullOf\n\n * @SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Any>\nSequence<T>.firstNotNullOf(transform: (T) -> R?): R {\n    return firstNotNullOfOrNull(transform) ?: throw\n    NoSuchElementException("No element of the sequence was transformed to a non-null value.")\n}\n\n * Returns the first non-null value produced by [transform] function being applied to elements of this sequence in\n * iteration order,\n * or `null` if no non-null value was produced.\n * The operation is _terminal_.\n * \n * @sample\nsamples.collections.Collections.Transformations.firstNotNullOf\n\n * @SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Any>

```

```

Sequence<T>.firstNotNullOrNull(transform: (T) -> R?): R? {
    for (element in this) {
        val result = transform(element)
        if (result != null) return result
    }
    return null
}
Returns the first element, or `null` if the sequence is empty.
The operation is _terminal_.
public fun <T> Sequence<T>.firstOrNull(): T? {
    val iterator = iterator()
    if (!iterator.hasNext()) return null
    return iterator.next()
}
Returns the first element matching the given [predicate], or `null` if element was not found.
The operation is _terminal_.
public inline fun <T> Sequence<T>.firstOrNull(predicate: (T) -> Boolean): T? {
    for (element in this) if (predicate(element)) return element
    return null
}
Returns first index of [element], or -1 if the sequence does not contain element.
The operation is _terminal_.
public fun <@kotlin.internal.OnlyInputTypes T> Sequence<T>.indexOf(element: T): Int {
    var index = 0
    for (item in this) {
        checkIndexOverflow(index)
        if (element == item) return index
        index++
    }
    return -1
}
Returns index of the first element matching the given [predicate], or -1 if the sequence does not contain such element.
The operation is _terminal_.
public inline fun <T> Sequence<T>.indexOfFirst(predicate: (T) -> Boolean): Int {
    var index = 0
    for (item in this) {
        checkIndexOverflow(index)
        if (predicate(item)) return index
        index++
    }
    return -1
}
Returns index of the last element matching the given [predicate], or -1 if the sequence does not contain such element.
The operation is _terminal_.
public inline fun <T> Sequence<T>.indexOfLast(predicate: (T) -> Boolean): Int {
    var lastIndex = -1
    var index = 0
    for (item in this) {
        checkIndexOverflow(index)
        if (predicate(item)) lastIndex = index
        index++
    }
    return lastIndex
}
Returns the last element.
The operation is _terminal_.
* @throws NoSuchElementException if the sequence is empty.
* @sample samples.collections.Collections.Elements.last
public fun <T> Sequence<T>.last(): T {
    val iterator = iterator()
    if (!iterator.hasNext()) throw NoSuchElementException("Sequence is empty.")
    var last = iterator.next()
    while (iterator.hasNext()) last = iterator.next()
    return last
}
Returns the last element matching the given [predicate].
The operation is _terminal_.
* @throws NoSuchElementException if no such element is found.
* @sample samples.collections.Collections.Elements.last
public inline fun <T> Sequence<T>.last(predicate: (T) -> Boolean): T {
    var last: T? = null
    var found = false
    for (element in this) {
        if (predicate(element)) {
            last = element
            found = true
        }
    }
    if (!found) throw NoSuchElementException("Sequence contains no element matching the predicate.")
    @Suppress("UNCHECKED_CAST") return last as T
}
Returns last index of [element], or -1 if the sequence does not contain element.
The operation is _terminal_.
public fun <@kotlin.internal.OnlyInputTypes T> Sequence<T>.lastIndexOf(element: T): Int {
    var lastIndex = -1
    var index = 0
    for (item in this) {
        checkIndexOverflow(index)
        if (element == item) lastIndex = index
        index++
    }
    return lastIndex
}
Returns the last element, or `null` if the sequence is empty.
The operation is _terminal_.
* @sample samples.collections.Collections.Elements.last
public fun <T> Sequence<T>.lastOrNull(): T? {
    val iterator = iterator()
    if (!iterator.hasNext()) return null
    var last = iterator.next()
    while (iterator.hasNext()) last = iterator.next()
    return last
}
Returns the last element matching the given [predicate], or `null` if no such element was found.
The operation is _terminal_.
* @sample samples.collections.Collections.Elements.last
public inline fun <T> Sequence<T>.lastOrNull(predicate: (T) -> Boolean): T? {
    var last: T? = null
    for (element in this) {
        if (predicate(element)) last = element
    }
    return last
}
Returns the single element, or throws an exception if the sequence is empty or has more than one element.
The operation is _terminal_.
public fun <T> Sequence<T>.single(): T {
    val iterator = iterator()
    if (!iterator.hasNext()) throw NoSuchElementException("Sequence is empty.")
    val single = iterator.next()
    if (iterator.hasNext()) throw IllegalArgumentException("Sequence has more than one element.")
    return single
}
Returns the single element matching the given [predicate], or throws exception if there is no or more than one matching element.
The operation is _terminal_.
public inline fun <T> Sequence<T>.single(predicate: (T) -> Boolean): T {
    var single: T? = null
    var found = false
    for

```

```

(element in this) {\n    if (predicate(element)) {\n        if (found) throw IllegalArgumentException("Sequence
contains more than one matching element.")\n        single = element\n        found = true\n    }\n    if
(!found) throw NoSuchElementException("Sequence contains no element matching the predicate.")\n
@Suppress("UNCHECKED_CAST")\n    return single as T\n}\n\n/**\n * Returns single element, or `null` if the
sequence is empty or has more than one element.\n *\n * The operation is _terminal_.\n */\npublic fun <T>
Sequence<T>.singleOrNull(): T? {\n    val iterator = iterator()\n    if (!iterator.hasNext())\n        return null\n    val
single = iterator.next()\n    if (iterator.hasNext())\n        return null\n    return single\n}\n\n/**\n * Returns the single
element matching the given [predicate], or `null` if element was not found or more than one element was found.\n
*\n * The operation is _terminal_.\n */\npublic inline fun <T> Sequence<T>.singleOrNull(predicate: (T) ->
Boolean): T? {\n    var single: T? = null\n    var found = false\n    for (element in this) {\n        if (predicate(element))
{\n            if (found) return null\n            single = element\n            found = true\n        }\n        if (!found) return
null\n        return single\n    }\n\n/**\n * Returns a sequence containing all elements except first [n] elements.\n *\n * The
operation is _intermediate_ and _stateless_.\n *\n * @throws IllegalArgumentException if [n] is negative.\n *\n *
@sample samples.collections.Collections.Transformations.drop\n */\npublic fun <T> Sequence<T>.drop(n: Int):
Sequence<T> {\n    require(n >= 0) { "Requested element count $n is less than zero." }\n    return when {\n        n
== 0 -> this\n        this is DropTakeSequence -> this.drop(n)\n        else -> DropSequence(this, n)\n    }\n}\n\n/**\n *
Returns a sequence containing all elements except first elements that satisfy the given [predicate].\n *\n * The
operation is _intermediate_ and _stateless_.\n *\n * @sample
samples.collections.Collections.Transformations.drop\n */\npublic fun <T> Sequence<T>.dropWhile(predicate: (T)
-> Boolean): Sequence<T> {\n    return DropWhileSequence(this, predicate)\n}\n\n/**\n * Returns a sequence
containing only elements matching the given [predicate].\n *\n * The operation is _intermediate_ and _stateless_.\n
*\n * @sample samples.collections.Collections.Filtering.filter\n */\npublic fun <T> Sequence<T>.filter(predicate:
(T) -> Boolean): Sequence<T> {\n    return FilteringSequence(this, true, predicate)\n}\n\n/**\n * Returns a sequence
containing only elements matching the given [predicate].\n *\n * @param [predicate] function that takes the index of an
element and the element itself\n * and returns the result of predicate evaluation on the element.\n *\n * The
operation is _intermediate_ and _stateless_.\n *\n * @sample
samples.collections.Collections.Filtering.filterIndexed\n */\npublic fun <T> Sequence<T>.filterIndexed(predicate:
(index: Int, T) -> Boolean): Sequence<T> {\n    // TODO: Rewrite with generalized MapFilterIndexingSequence\n
return TransformingSequence(FilteringSequence(IndexingSequence(this), true, { predicate(it.index, it.value) }), {
it.value })\n}\n\n/**\n * Appends all elements matching the given [predicate] to the given [destination].\n *\n * @param
[predicate] function that takes the index of an element and the element itself\n * and returns the result of predicate
evaluation on the element.\n *\n * The operation is _terminal_.\n *\n * @sample
samples.collections.Collections.Filtering.filterIndexedTo\n */\npublic inline fun <T, C : MutableCollection<in T>>
Sequence<T>.filterIndexedTo(destination: C, predicate: (index: Int, T) -> Boolean): C {\n    forEachIndexed {\n
index, element ->\n        if (predicate(index, element)) destination.add(element)\n    }\n    return
destination\n}\n\n/**\n * Returns a sequence containing all elements that are instances of specified type parameter
R.\n *\n * The operation is _intermediate_ and _stateless_.\n *\n * @sample
samples.collections.Collections.Filtering.filterIsInstance\n */\npublic inline fun <reified R>
Sequence<*>.filterIsInstance(): Sequence<@kotlin.internal.NoInfer R> {\n
@Suppress("UNCHECKED_CAST")\n    return filter { it is R } as Sequence<R>\n}\n\n/**\n * Appends all
elements that are instances of specified type parameter R to the given [destination].\n *\n * The operation is
_terminal_.\n *\n * @sample samples.collections.Collections.Filtering.filterIsInstanceTo\n */\npublic inline fun
<reified R, C : MutableCollection<in R>> Sequence<*>.filterIsInstanceTo(destination: C): C {\n    for (element in
this) if (element is R) destination.add(element)\n    return destination\n}\n\n/**\n * Returns a sequence containing
all elements not matching the given [predicate].\n *\n * The operation is _intermediate_ and _stateless_.\n *\n *
@sample samples.collections.Collections.Filtering.filter\n */\npublic fun <T> Sequence<T>.filterNot(predicate: (T)
-> Boolean): Sequence<T> {\n    return FilteringSequence(this, false, predicate)\n}\n\n/**\n * Returns a sequence
containing all elements that are not `null`. \n *\n * The operation is _intermediate_ and _stateless_.\n *\n * @sample

```

```

samples.collections.Collections.Filtering.filterNotNull\n */\npublic fun <T : Any> Sequence<T?>.filterNotNull():
Sequence<T> {\n    @Suppress("UNCHECKED_CAST")\n    return filterNot { it == null } as
Sequence<T>}\n\n/**\n * Appends all elements that are not `null` to the given [destination].\n */\n * The operation
is _terminal_.\n */\n * @sample samples.collections.Collections.Filtering.filterNotNullTo\n */\npublic fun <C :
MutableCollection<in T>, T : Any> Sequence<T?>.filterNotNullTo(destination: C): C {\n    for (element in this) if
(element != null) destination.add(element)\n    return destination}\n\n/**\n * Appends all elements not matching
the given [predicate] to the given [destination].\n */\n * The operation is _terminal_.\n */\n * @sample
samples.collections.Collections.Filtering.filterTo\n */\npublic inline fun <T, C : MutableCollection<in T>>
Sequence<T>.filterNotTo(destination: C, predicate: (T) -> Boolean): C {\n    for (element in this) if
(!predicate(element)) destination.add(element)\n    return destination}\n\n/**\n * Appends all elements matching
the given [predicate] to the given [destination].\n */\n * The operation is _terminal_.\n */\n * @sample
samples.collections.Collections.Filtering.filterTo\n */\npublic inline fun <T, C : MutableCollection<in T>>
Sequence<T>.filterTo(destination: C, predicate: (T) -> Boolean): C {\n    for (element in this) if (predicate(element))
destination.add(element)\n    return destination}\n\n/**\n * Returns a sequence containing first [n] elements.\n */\n
* The operation is _intermediate_ and _stateless_.\n */\n * @throws IllegalArgumentException if [n] is negative.\n */
\n * @sample samples.collections.Collections.Transformations.take\n */\npublic fun <T> Sequence<T>.take(n: Int):
Sequence<T> {\n    require(n >= 0) { "Requested element count $n is less than zero." }\n    return when {\n        n
== 0 -> emptySequence()\n        this is DropTakeSequence -> this.take(n)\n        else -> TakeSequence(this, n)\n
}\n}\n\n/**\n * Returns a sequence containing first elements satisfying the given [predicate].\n */\n * The operation
is _intermediate_ and _stateless_.\n */\n * @sample samples.collections.Collections.Transformations.take\n
*/\npublic fun <T> Sequence<T>.takeWhile(predicate: (T) -> Boolean): Sequence<T> {\n    return
TakeWhileSequence(this, predicate)\n}\n\n/**\n * Returns a sequence that yields elements of this sequence sorted
according to their natural sort order.\n */\n * The sort is _stable_. It means that equal elements preserve their order
relative to each other after sorting.\n */\n * The operation is _intermediate_ and _stateful_.\n */\npublic fun <T :
Comparable<T>> Sequence<T>.sorted(): Sequence<T> {\n    return object : Sequence<T> {\n        override fun
iterator(): Iterator<T> {\n            val sortedList = this@sorted.toMutableList()\n            sortedList.sort()\n
return sortedList.iterator()\n        }\n    }\n}\n\n/**\n * Returns a sequence that yields elements of this sequence
sorted according to natural sort order of the value returned by specified [selector] function.\n */\n * The sort is
_stable_. It means that equal elements preserve their order relative to each other after sorting.\n */\n * The operation
is _intermediate_ and _stateful_.\n */\n * @sample samples.collections.Collections.Sorting.sortedBy\n */\npublic
inline fun <T, R : Comparable<R>> Sequence<T>.sortedBy(crossinline selector: (T) -> R?): Sequence<T> {\n
return sortedWith(compareBy(selector))}\n}\n\n/**\n * Returns a sequence that yields elements of this sequence
sorted descending according to natural sort order of the value returned by specified [selector] function.\n */\n * The
sort is _stable_. It means that equal elements preserve their order relative to each other after sorting.\n */\n * The
operation is _intermediate_ and _stateful_.\n */\npublic inline fun <T, R : Comparable<R>>
Sequence<T>.sortedByDescending(crossinline selector: (T) -> R?): Sequence<T> {\n    return
sortedWith(compareByDescending(selector))}\n}\n\n/**\n * Returns a sequence that yields elements of this sequence
sorted descending according to their natural sort order.\n */\n * The sort is _stable_. It means that equal elements
preserve their order relative to each other after sorting.\n */\n * The operation is _intermediate_ and _stateful_.\n
*/\npublic fun <T : Comparable<T>> Sequence<T>.sortedDescending(): Sequence<T> {\n    return
sortedWith(reverseOrder())}\n}\n\n/**\n * Returns a sequence that yields elements of this sequence sorted according
to the specified [comparator].\n */\n * The sort is _stable_. It means that equal elements preserve their order relative
to each other after sorting.\n */\n * The operation is _intermediate_ and _stateful_.\n */\npublic fun <T>
Sequence<T>.sortedWith(comparator: Comparator<in T>): Sequence<T> {\n    return object : Sequence<T> {\n
override fun iterator(): Iterator<T> {\n        val sortedList = this@sortedWith.toMutableList()\n
sortedList.sortWith(comparator)\n        return sortedList.iterator()\n    }\n}\n}\n\n/**\n * Returns a [Map]
containing key-value pairs provided by [transform] function\n */\n * applied to elements of the given sequence.\n */\n *
\n * If any of two pairs would have the same key the last one gets added to the map.\n */\n * The returned map preserves

```

the entry iteration order of the original sequence.

```

    * The operation is _terminal_.
    * @sample
samples.collections.Collections.Transformations.associate
    * \n public inline fun <T, K, V>
Sequence<T>.associate(transform: (T) -> Pair<K, V>): Map<K, V> {
    * \n return associateTo(LinkedHashMap<K,
    * V>(), transform)\n}
    * \n /**
    * Returns a [Map] containing the elements from the given sequence indexed by the
    * key
    * returned from [keySelector] function applied to each element.
    * \n * \n * If any two elements would have the
    * same key returned by [keySelector] the last one gets added to the map.
    * \n * \n * The returned map preserves the entry
    * iteration order of the original sequence.
    * \n * \n * The operation is _terminal_.
    * \n * \n * @sample
samples.collections.Collections.Transformations.associateBy
    * \n public inline fun <T, K>
Sequence<T>.associateBy(keySelector: (T) -> K): Map<K, T> {
    * \n return associateByTo(LinkedHashMap<K,
    * T>(), keySelector)\n}
    * \n /**
    * Returns a [Map] containing the values provided by [valueTransform] and indexed
    * by [keySelector] functions applied to elements of the given sequence.
    * \n * \n * If any two elements would have the
    * same key returned by [keySelector] the last one gets added to the map.
    * \n * \n * The returned map preserves the entry
    * iteration order of the original sequence.
    * \n * \n * The operation is _terminal_.
    * \n * \n * @sample
samples.collections.Collections.Transformations.associateByWithValueTransform
    * \n public inline fun <T, K, V>
Sequence<T>.associateBy(keySelector: (T) -> K, valueTransform: (T) -> V): Map<K, V> {
    * \n return
    * associateByTo(LinkedHashMap<K, V>(), keySelector, valueTransform)\n}
    * \n /**
    * Populates and returns the
    * [destination] mutable map with key-value pairs,
    * \n * where key is provided by the [keySelector] function applied to
    * each element of the given sequence
    * \n * and value is the element itself.
    * \n * \n * If any two elements would have the
    * same key returned by [keySelector] the last one gets added to the map.
    * \n * \n * The operation is _terminal_.
    * \n * \n * @sample
samples.collections.Collections.Transformations.associateByTo
    * \n public inline fun <T, K, M :
MutableMap<in K, in T>> Sequence<T>.associateByTo(destination: M, keySelector: (T) -> K): M {
    * \n for
    * (element in this) {
    * \n destination.put(keySelector(element), element)\n }
    * \n return destination\n}
    * \n /**
    * Populates and returns the [destination] mutable map with key-value pairs,
    * \n * where key is provided by the
    * [keySelector] function and
    * \n * and value is provided by the [valueTransform] function applied to elements of the
    * given sequence.
    * \n * \n * If any two elements would have the same key returned by [keySelector] the last one gets
    * added to the map.
    * \n * \n * The operation is _terminal_.
    * \n * \n * @sample
samples.collections.Collections.Transformations.associateByToWithValueTransform
    * \n public inline fun <T, K,
V, M : MutableMap<in K, in V>> Sequence<T>.associateByTo(destination: M, keySelector: (T) -> K,
    * valueTransform: (T) -> V): M {
    * \n for (element in this) {
    * \n destination.put(keySelector(element),
    * valueTransform(element))\n }
    * \n return destination\n}
    * \n /**
    * Populates and returns the [destination] mutable
    * map with key-value pairs
    * \n * provided by [transform] function applied to each element of the given sequence.
    * \n * \n * If any of two pairs would have the same key the last one gets added to the map.
    * \n * \n * The operation is
    * _terminal_.
    * \n * \n * @sample
samples.collections.Collections.Transformations.associateTo
    * \n public inline fun
<T, K, V, M : MutableMap<in K, in V>> Sequence<T>.associateTo(destination: M, transform: (T) -> Pair<K, V>):
    * M {
    * \n for (element in this) {
    * \n destination += transform(element)\n }
    * \n return destination\n}
    * \n /**
    * Returns a [Map] where keys are elements from the given sequence and values are
    * produced by the
    * [valueSelector] function applied to each element.
    * \n * \n * If any two elements are equal, the last one gets added to
    * the map.
    * \n * \n * The returned map preserves the entry iteration order of the original sequence.
    * \n * \n * The
    * operation is _terminal_.
    * \n * \n * @sample
samples.collections.Collections.Transformations.associateWith
    * \n public inline fun
    * \n @SinceKotlin("1.3")
    * \n public inline fun <K, V> Sequence<K>.associateWith(valueSelector: (K) -> V):
    * Map<K, V> {
    * \n val result = LinkedHashMap<K, V>()\n return associateWithTo(result,
    * valueSelector)\n}
    * \n /**
    * Populates and returns the [destination] mutable map with key-value pairs for each
    * element of the given sequence,
    * \n * where key is the element itself and value is provided by the [valueSelector]
    * function applied to that key.
    * \n * \n * If any two elements are equal, the last one overwrites the former value in the
    * map.
    * \n * \n * The operation is _terminal_.
    * \n * \n * @sample
samples.collections.Collections.Transformations.associateWithTo
    * \n @SinceKotlin("1.3")
    * \n public inline fun
<K, V, M : MutableMap<in K, in V>> Sequence<K>.associateWithTo(destination: M, valueSelector: (K) -> V): M {
    * \n for (element in this) {
    * \n destination.put(element, valueSelector(element))\n }
    * \n return

```

```

destination\n}\n\n/**\n * Appends all elements to the given [destination] collection.\n *\n * The operation is
_terminal_.\n */\npublic fun <T, C : MutableCollection<in T>> Sequence<T>.toCollection(destination: C): C {\n
for (item in this) {\n    destination.add(item)\n }\n return destination\n}\n\n/**\n * Returns a new [HashSet] of
all elements.\n *\n * The operation is _terminal_.\n */\npublic fun <T> Sequence<T>.toHashSet(): HashSet<T> {\n
return toCollection(HashSet<T>())\n}\n\n/**\n * Returns a [List] containing all elements.\n *\n * The operation is
_terminal_.\n */\npublic fun <T> Sequence<T>.toList(): List<T> {\n    return
this.toMutableList().optimizeReadOnlyList()\n}\n\n/**\n * Returns a new [MutableList] filled with all elements of
this sequence.\n *\n * The operation is _terminal_.\n */\npublic fun <T> Sequence<T>.toMutableList():
MutableList<T> {\n    return toCollection(ArrayList<T>())\n}\n\n/**\n * Returns a [Set] of all elements.\n *\n *
The returned set preserves the element iteration order of the original sequence.\n *\n * The operation is
_terminal_.\n */\npublic fun <T> Sequence<T>.toSet(): Set<T> {\n    return
toCollection(LinkedHashSet<T>()).optimizeReadOnlySet()\n}\n\n/**\n * Returns a single sequence of all elements
from results of [transform] function being invoked on each element of original sequence.\n *\n * The operation is
_intermediate_ and _stateless_.\n *\n * @sample samples.collections.Collections.Transformations.flatMap\n
*/\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIterable")\npublic fun <T, R>
Sequence<T>.flatMap(transform: (T) -> Iterable<R>): Sequence<R> {\n    return FlatteningSequence(this,
transform, Iterable<R>::iterator)\n}\n\n/**\n * Returns a single sequence of all elements from results of [transform]
function being invoked on each element of original sequence.\n *\n * The operation is _intermediate_ and
_stateless_.\n *\n * @sample samples.collections.Collections.Transformations.flatMap\n */\n\npublic fun <T, R>
Sequence<T>.flatMap(transform: (T) -> Sequence<R>): Sequence<R> {\n    return FlatteningSequence(this,
transform, Sequence<R>::iterator)\n}\n\n/**\n * Returns a single sequence of all elements yielded from results of
[transform] function being invoked on each element\n * and its index in the original sequence.\n *\n * The operation
is _intermediate_ and _stateless_.\n *\n * @sample
samples.collections.Collections.Transformations.flatMapIndexed\n
*/\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterable")\npublic fun <T, R>
Sequence<T>.flatMapIndexed(transform: (index: Int, T) -> Iterable<R>): Sequence<R> {\n    return
flatMapIndexed(this, transform, Iterable<R>::iterator)\n}\n\n/**\n * Returns a single sequence of all elements
yielded from results of [transform] function being invoked on each element\n * and its index in the original
sequence.\n *\n * The operation is _intermediate_ and _stateless_.\n *\n * @sample
samples.collections.Collections.Transformations.flatMapIndexed\n
*/\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedSequence")\npublic fun <T, R>
Sequence<T>.flatMapIndexed(transform: (index: Int, T) -> Sequence<R>): Sequence<R> {\n    return
flatMapIndexed(this, transform, Sequence<R>::iterator)\n}\n\n/**\n * Appends all elements yielded from results of
[transform] function being invoked on each element\n * and its index in the original sequence, to the given
[destination].\n *\n * The operation is _terminal_.\n
*/\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedIterableTo")\n@kotlin.internal.InlineOnly\npublic
inline fun <T, R, C : MutableCollection<in R>> Sequence<T>.flatMapIndexedTo(destination: C, transform:
(index: Int, T) -> Iterable<R>): C {\n    var index = 0\n    for (element in this) {\n        val list =
transform(checkIndexOverflow(index++), element)\n        destination.addAll(list)\n    }\n    return
destination\n}\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each
element\n * and its index in the original sequence, to the given [destination].\n *\n * The operation is _terminal_.\n
*/\n\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIndexedSequenceTo")\n@kotlin.internal.InlineOnly\npublic
inline fun <T, R, C : MutableCollection<in R>> Sequence<T>.flatMapIndexedTo(destination: C, transform:

```

```
(index: Int, T) -> Sequence<R>: C {\n    var index = 0\n    for (element in this) {\n        val list =  
transform(checkIndexOverflow(index++), element)\n        destination.addAll(list)\n    }\n    return  
destination\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each  
element of original sequence, to the given [destination].\n *\n * The operation is _terminal_.\n *\n * @since Kotlin("1.4")\n */\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.jvm.JvmName("flatMapIterableTo")\npublic inline fun <T, R, C :  
MutableCollection<in R>> Sequence<T>.flatMapTo(destination: C, transform: (T) -> Iterable<R>): C {\n    for  
(element in this) {\n        val list = transform(element)\n        destination.addAll(list)\n    }\n    return  
destination\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each  
element of original sequence, to the given [destination].\n *\n * The operation is _terminal_.\n *\n * @public inline fun  
<T, R, C : MutableCollection<in R>> Sequence<T>.flatMapTo(destination: C, transform: (T) -> Sequence<R>): C  
{\n    for (element in this) {\n        val list = transform(element)\n        destination.addAll(list)\n    }\n    return  
destination\n}\n\n/**\n * Groups elements of the original sequence by the key returned by the given [keySelector]  
function\n * applied to each element and returns a map where each group key is associated with a list of  
corresponding elements.\n *\n * The returned map preserves the entry iteration order of the keys produced from the  
original sequence.\n *\n * The operation is _terminal_.\n *\n * @sample  
samples.collections.Collections.Transformations.groupBy\n */\npublic inline fun <T, K>  
Sequence<T>.groupBy(keySelector: (T) -> K): Map<K, List<T>> {\n    return groupByTo(LinkedHashMap<K,  
MutableList<T>>(), keySelector)\n}\n\n/**\n * Groups values returned by the [valueTransform] function applied to  
each element of the original sequence\n * by the key returned by the given [keySelector] function applied to the  
element\n * and returns a map where each group key is associated with a list of corresponding values.\n *\n * The  
returned map preserves the entry iteration order of the keys produced from the original sequence.\n *\n * The  
operation is _terminal_.\n *\n * @sample  
samples.collections.Collections.Transformations.groupByKeysAndValues\n */\npublic inline fun <T, K, V>  
Sequence<T>.groupBy(keySelector: (T) -> K, valueTransform: (T) -> V): Map<K, List<V>> {\n    return  
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups elements  
of the original sequence by the key returned by the given [keySelector] function\n * applied to each element and  
puts to the [destination] map each group key associated with a list of corresponding elements.\n *\n * @return The  
[destination] map.\n *\n * The operation is _terminal_.\n *\n * @sample  
samples.collections.Collections.Transformations.groupBy\n */\npublic inline fun <T, K, M : MutableMap<in K,  
MutableList<T>>> Sequence<T>.groupByTo(destination: M, keySelector: (T) -> K): M {\n    for (element in this)  
{\n        val key = keySelector(element)\n        val list = destination.getOrPut(key) { ArrayList<T>() }\n        list.add(element)\n    }\n    return destination\n}\n\n/**\n * Groups values returned by the [valueTransform] function  
applied to each element of the original sequence\n * by the key returned by the given [keySelector] function applied  
to the element\n * and puts to the [destination] map each group key associated with a list of corresponding values.\n *\n * @return The [destination] map.\n *\n * The operation is _terminal_.\n *\n * @sample  
samples.collections.Collections.Transformations.groupByKeysAndValues\n */\npublic inline fun <T, K, V, M :  
MutableMap<in K, MutableList<V>>> Sequence<T>.groupByTo(destination: M, keySelector: (T) -> K,  
valueTransform: (T) -> V): M {\n    for (element in this) {\n        val key = keySelector(element)\n        val list =  
destination.getOrPut(key) { ArrayList<V>() }\n        list.add(valueTransform(element))\n    }\n    return  
destination\n}\n\n/**\n * Creates a [Grouping] source from a sequence to be used later with one of group-and-fold  
operations\n * using the specified [keySelector] function to extract a key from each element.\n *\n * The operation is  
_intermediate_ and _stateless_.\n *\n * @sample samples.collections.Grouping.groupingByEachCount\n */\n@SinceKotlin("1.1")\npublic inline fun <T, K> Sequence<T>.groupingBy(crossinline keySelector: (T) -> K):  
Grouping<T, K> {\n    return object : Grouping<T, K> {\n        override fun sourceIterator(): Iterator<T> =  
this@groupingBy.iterator()\n        override fun keyOf(element: T): K = keySelector(element)\n    }\n}\n\n/**\n * Returns a sequence containing the results of applying the given [transform] function\n * to each element in the  
original sequence.\n *\n * The operation is _intermediate_ and _stateless_.\n *\n * @sample
```

```

samples.collections.Collections.Transformations.map\n *
\npublic fun <T, R> Sequence<T>.map(transform: (T) ->
R): Sequence<R> {\n    return TransformingSequence(this, transform)\n}\n\n/**\n * Returns a sequence containing
the results of applying the given [transform] function\n * to each element and its index in the original sequence.\n *
@param [transform] function that takes the index of an element and the element itself\n * and returns the result of
the transform applied to the element.\n *\n * The operation is _intermediate_ and _stateless_.\n */\npublic fun <T,
R> Sequence<T>.mapIndexed(transform: (index: Int, T) -> R): Sequence<R> {\n    return
TransformingIndexedSequence(this, transform)\n}\n\n/**\n * Returns a sequence containing only the non-null
results of applying the given [transform] function\n * to each element and its index in the original sequence.\n *
@param [transform] function that takes the index of an element and the element itself\n * and returns the result of
the transform applied to the element.\n *\n * The operation is _intermediate_ and _stateless_.\n */\npublic fun <T, R
: Any> Sequence<T>.mapIndexedNotNull(transform: (index: Int, T) -> R?): Sequence<R> {\n    return
TransformingIndexedSequence(this, transform).filterNotNull()\n}\n\n/**\n * Applies the given [transform] function
to each element and its index in the original sequence\n * and appends only the non-null results to the given
[destination].\n *
@param [transform] function that takes the index of an element and the element itself\n * and
returns the result of the transform applied to the element.\n *\n * The operation is _terminal_.\n */\npublic inline fun
<T, R : Any, C : MutableCollection<in R>> Sequence<T>.mapIndexedNotNullTo(destination: C, transform: (index:
Int, T) -> R?): C {\n    forEachIndexed { index, element -> transform(index, element)?.let { destination.add(it) } }\n
return destination\n}\n\n/**\n * Applies the given [transform] function to each element and its index in the original
sequence\n * and appends the results to the given [destination].\n *
@param [transform] function that takes the
index of an element and the element itself\n * and returns the result of the transform applied to the element.\n *\n *
The operation is _terminal_.\n */\npublic inline fun <T, R, C : MutableCollection<in R>>
Sequence<T>.mapIndexedTo(destination: C, transform: (index: Int, T) -> R): C {\n    var index = 0\n    for (item in
this)\n        destination.add(transform(checkIndexOverflow(index++), item))\n    return destination\n}\n\n/**\n * Returns a sequence containing only the non-null results of applying the given [transform] function\n * to each
element in the original sequence.\n *\n * The operation is _intermediate_ and _stateless_.\n *\n * @sample
samples.collections.Collections.Transformations.mapNotNull\n */\npublic fun <T, R : Any>
Sequence<T>.mapNotNull(transform: (T) -> R?): Sequence<R> {\n    return TransformingSequence(this,
transform).filterNotNull()\n}\n\n/**\n * Applies the given [transform] function to each element in the original
sequence\n * and appends only the non-null results to the given [destination].\n *\n * The operation is _terminal_.\n */\npublic inline fun <T, R : Any, C : MutableCollection<in R>> Sequence<T>.mapNotNullTo(destination: C,
transform: (T) -> R?): C {\n    forEach { element -> transform(element)?.let { destination.add(it) } }\n    return
destination\n}\n\n/**\n * Applies the given [transform] function to each element of the original sequence\n * and
appends the results to the given [destination].\n *\n * The operation is _terminal_.\n */\npublic inline fun <T, R, C :
MutableCollection<in R>> Sequence<T>.mapTo(destination: C, transform: (T) -> R): C {\n    for (item in this)\n        destination.add(transform(item))\n    return destination\n}\n\n/**\n * Returns a sequence that wraps each element of
the original sequence\n * into an [IndexedValue] containing the index of that element and the element itself.\n *\n *
The operation is _intermediate_ and _stateless_.\n */\npublic fun <T> Sequence<T>.withIndex():
Sequence<IndexedValue<T>> {\n    return IndexingSequence(this)\n}\n\n/**\n * Returns a sequence containing
only distinct elements from the given sequence.\n *\n * Among equal elements of the given sequence, only the first
one will be present in the resulting sequence.\n *\n * The elements in the resulting sequence are in the same order as
they were in the source sequence.\n *\n * The operation is _intermediate_ and _stateful_.\n *\n * @sample
samples.collections.Collections.Transformations.distinctAndDistinctBy\n */\npublic fun <T>
Sequence<T>.distinct(): Sequence<T> {\n    return this.distinctBy { it }\n}\n\n/**\n * Returns a sequence
containing only elements from the given sequence\n * having distinct keys returned by the given [selector]
function.\n *\n * Among elements of the given sequence with equal keys, only the first one will be present in the
resulting sequence.\n *\n * The elements in the resulting sequence are in the same order as they were in the source
sequence.\n *\n * The operation is _intermediate_ and _stateful_.\n *\n * @sample
samples.collections.Collections.Transformations.distinctAndDistinctBy\n */\npublic fun <T, K>

```

Sequence<T>.distinctBy(selector: (T) -> K): Sequence<T> {\n return DistinctSequence(this, selector)\n}\n\n/\*\*\n \* Returns a new [MutableSet] containing all distinct elements from the given sequence.\n \* \n \* The returned set preserves the element iteration order of the original sequence.\n \* \n \* The operation is \_terminal\_.\n \*/\npublic fun <T> Sequence<T>.toMutableSet(): MutableSet<T> {\n val set = LinkedHashSet<T>()\n for (item in this) set.add(item)\n return set\n}\n\n/\*\*\n \* Returns `true` if all elements match the given [predicate].\n \* \n \* The operation is \_terminal\_.\n \* \n \* @sample samples.collections.Collections.Aggregates.all\n \*/\npublic inline fun <T> Sequence<T>.all(predicate: (T) -> Boolean): Boolean {\n for (element in this) if (!predicate(element)) return false\n return true\n}\n\n/\*\*\n \* Returns `true` if sequence has at least one element.\n \* \n \* The operation is \_terminal\_.\n \* \n \* @sample samples.collections.Collections.Aggregates.any\n \*/\npublic inline fun <T> Sequence<T>.any(): Boolean {\n return iterator().hasNext()\n}\n\n/\*\*\n \* Returns `true` if at least one element matches the given [predicate].\n \* \n \* The operation is \_terminal\_.\n \* \n \* @sample samples.collections.Collections.Aggregates.anyWithPredicate\n \*/\npublic inline fun <T> Sequence<T>.any(predicate: (T) -> Boolean): Boolean {\n for (element in this) if (predicate(element)) return true\n return false\n}\n\n/\*\*\n \* Returns the number of elements in this sequence.\n \* \n \* The operation is \_terminal\_.\n \*/\npublic inline fun <T> Sequence<T>.count(): Int {\n var count = 0\n for (element in this) checkCountOverflow(++count)\n return count\n}\n\n/\*\*\n \* Returns the number of elements matching the given [predicate].\n \* \n \* The operation is \_terminal\_.\n \*/\npublic inline fun <T> Sequence<T>.count(predicate: (T) -> Boolean): Int {\n var count = 0\n for (element in this) if (predicate(element)) checkCountOverflow(++count)\n return count\n}\n\n/\*\*\n \* Accumulates value starting with [initial] value and applying [operation] from left to right\n \* to current accumulator value and each element.\n \* \n \* Returns the specified [initial] value if the sequence is empty.\n \* \n \* @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.\n \* \n \* The operation is \_terminal\_.\n \*/\npublic inline fun <T, R> Sequence<T>.fold(initial: R, operation: (acc: R, T) -> R): R {\n var accumulator = initial\n for (element in this) accumulator = operation(accumulator, element)\n return accumulator\n}\n\n/\*\*\n \* Accumulates value starting with [initial] value and applying [operation] from left to right\n \* to current accumulator value and each element with its index in the original sequence.\n \* \n \* Returns the specified [initial] value if the sequence is empty.\n \* \n \* @param [operation] function that takes the index of an element, current accumulator value\n \* and the element itself, and calculates the next accumulator value.\n \* \n \* The operation is \_terminal\_.\n \*/\npublic inline fun <T, R> Sequence<T>.foldIndexed(initial: R, operation: (index: Int, acc: R, T) -> R): R {\n var index = 0\n var accumulator = initial\n for (element in this) accumulator = operation(checkIndexOverflow(index++), accumulator, element)\n return accumulator\n}\n\n/\*\*\n \* Performs the given [action] on each element.\n \* \n \* The operation is \_terminal\_.\n \*/\npublic inline fun <T> Sequence<T>.forEach(action: (T) -> Unit): Unit {\n for (element in this) action(element)\n}\n\n/\*\*\n \* Performs the given [action] on each element, providing sequential index with the element.\n \* \n \* @param [action] function that takes the index of an element and the element itself\n \* and performs the action on the element.\n \* \n \* The operation is \_terminal\_.\n \*/\npublic inline fun <T> Sequence<T>.forEachIndexed(action: (index: Int, T) -> Unit): Unit {\n var index = 0\n for (item in this) action(checkIndexOverflow(index++), item)\n}\n\n@Deprecated("Use maxOrNull instead.")\nReplaceWith("this.maxOrNull()")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")\n@SinceKotlin("1.1")\npublic fun Sequence<Double>.max(): Double? {\n return maxOrNull()\n}\n\n@Deprecated("Use maxOrNull instead.")\nReplaceWith("this.maxOrNull()")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")\n@SinceKotlin("1.1")\npublic fun Sequence<Float>.max(): Float? {\n return maxOrNull()\n}\n\n@Deprecated("Use maxOrNull()")\nReplaceWith("this.maxOrNull()")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")\npublic fun <T : Comparable<T>> Sequence<T>.max(): T? {\n return maxOrNull()\n}\n\n@Deprecated("Use maxByOrNull instead.")\nReplaceWith("this.maxByOrNull(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")\npublic inline fun <T, R : Comparable<R>> Sequence<T>.maxBy(selector: (T) ->

```

R): T? {
    return maxByOrNull(selector)\n}\n\n/**
 * Returns the first element yielding the largest value of the
 * given function or `null` if there are no elements.\n *
 * The operation is _terminal_.\n *
 * @sample
 * samples.collections.Collections.Aggregates.maxByOrNull\n *
 * @SinceKotlin("1.4")\npublic inline fun <T, R :
 * Comparable<R>> Sequence<T>.maxByOrNull(selector: (T) -> R): T? {
    val iterator = iterator()\n    if
    (!iterator.hasNext()) return null\n    var maxElem = iterator.next()\n    if (!iterator.hasNext()) return maxElem\n    var
    maxElem = selector(maxElem)\n    do {
        val e = iterator.next()\n        val v = selector(e)\n        if (maxElem <
        v) {
            maxElem = e\n            maxElem = v\n        }\n    } while (iterator.hasNext())\n    return
    maxElem\n}\n\n/**
 * Returns the largest value among all values produced by [selector] function\n *
 * applied to each element in the sequence.\n *
 * If any of values produced by [selector] function is `NaN`, the returned result
 * is `NaN`.\n *
 * @throws NoSuchElementException if the sequence is empty.\n *
 * The operation is
    _terminal_.\n
 *
 * @SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
 * ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Sequence<T>.maxOf(selector: (T) ->
 * Double): Double {
    val iterator = iterator()\n    if (!iterator.hasNext()) throw NoSuchElementException()\n    var
    maxElem = selector(iterator.next())\n    while (iterator.hasNext()) {
        val v = selector(iterator.next())\n
        maxElem = maxOf(maxElem, v)\n    }\n    return maxElem\n}\n\n/**
 * Returns the largest value among all
 * values produced by [selector] function\n *
 * applied to each element in the sequence.\n *
 * If any of values
 * produced by [selector] function is `NaN`, the returned result is `NaN`.\n *
 * @throws NoSuchElementException
 * if the sequence is empty.\n *
 * The operation is _terminal_.\n
 *
 * @SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
 * ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Sequence<T>.maxOf(selector: (T) ->
 * Float): Float {
    val iterator = iterator()\n    if (!iterator.hasNext()) throw NoSuchElementException()\n    var
    maxElem = selector(iterator.next())\n    while (iterator.hasNext()) {
        val v = selector(iterator.next())\n
        maxElem = maxOf(maxElem, v)\n    }\n    return maxElem\n}\n\n/**
 * Returns the largest value among all
 * values produced by [selector] function\n *
 * applied to each element in the sequence.\n *
 * @throws
 * NoSuchElementException if the sequence is empty.\n *
 * The operation is _terminal_.\n
 *
 * @SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
 * ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Comparable<R>>
 * Sequence<T>.maxOf(selector: (T) -> R): R {
    val iterator = iterator()\n    if (!iterator.hasNext()) throw
    NoSuchElementException()\n    var maxElem = selector(iterator.next())\n    while (iterator.hasNext()) {
        val v
        = selector(iterator.next())\n        if (maxElem < v) {
            maxElem = v\n        }\n    }\n    return
    maxElem\n}\n\n/**
 * Returns the largest value among all values produced by [selector] function\n *
 * applied to
 * each element in the sequence or `null` if there are no elements.\n *
 * If any of values produced by [selector]
 * function is `NaN`, the returned result is `NaN`.\n *
 * The operation is _terminal_.\n
 *
 * @SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
 * ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Sequence<T>.maxOfOrNull(selector:
 * (T) -> Double): Double? {
    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var maxElem =
    selector(iterator.next())\n    while (iterator.hasNext()) {
        val v = selector(iterator.next())\n
        maxElem =
        maxOf(maxElem, v)\n    }\n    return maxElem\n}\n\n/**
 * Returns the largest value among all values produced
 * by [selector] function\n *
 * applied to each element in the sequence or `null` if there are no elements.\n *
 * If any of
 * values produced by [selector] function is `NaN`, the returned result is `NaN`.\n *
 * The operation is _terminal_.\n
 *
 * @SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
 * ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T> Sequence<T>.maxOfOrNull(selector:
 * (T) -> Float): Float? {
    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var maxElem =
    selector(iterator.next())\n    while (iterator.hasNext()) {
        val v = selector(iterator.next())\n
        maxElem =
        maxOf(maxElem, v)\n    }\n    return maxElem\n}\n\n/**
 * Returns the largest value among all values produced
 * by [selector] function\n *
 * applied to each element in the sequence or `null` if there are no elements.\n *
 * The
 * operation is _terminal_.\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Comparable<R>>
Sequence<T>.maxOfOrNull(selector: (T) -> R): R? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return
null\n    var max Value = selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v =
selector(iterator.next())\n        if (max Value < v) {\n            max Value = v\n        }\n    }\n    return
max Value\n}\n\n/**\n * Returns the largest value according to the provided [comparator]\n * among all values
produced by [selector] function applied to each element in the sequence.\n * \n * @throws
NoSuchElementException if the sequence is empty.\n * \n * The operation is _terminal_.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R>
Sequence<T>.maxOfWith(comparator: Comparator<in R>, selector: (T) -> R): R {\n    val iterator = iterator()\n    if
(!iterator.hasNext()) throw NoSuchElementException()\n    var max Value = selector(iterator.next())\n    while
(iterator.hasNext()) {\n        val v = selector(iterator.next())\n        if (comparator.compare(max Value, v) < 0) {\n
            max Value = v\n        }\n    }\n    return max Value\n}\n\n/**\n * Returns the largest value according to the provided
[comparator]\n * among all values produced by [selector] function applied to each element in the sequence or `null`
if there are no elements.\n * \n * The operation is _terminal_.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R>
Sequence<T>.maxOfWithOrNull(comparator: Comparator<in R>, selector: (T) -> R): R? {\n    val iterator =
iterator()\n    if (!iterator.hasNext()) return null\n    var max Value = selector(iterator.next())\n    while
(iterator.hasNext()) {\n        val v = selector(iterator.next())\n        if (comparator.compare(max Value, v) < 0) {\n
            max Value = v\n        }\n    }\n    return max Value\n}\n\n/**\n * Returns the largest element or `null` if there are no
elements.\n * \n * If any of elements is `NaN` returns `NaN`.\n * \n * The operation is _terminal_.\n
*\n@SinceKotlin("1.4")\npublic fun Sequence<Double>.maxOrNull(): Double? {\n    val iterator = iterator()\n
if (!iterator.hasNext()) return null\n    var max = iterator.next()\n    while (iterator.hasNext()) {\n        val e =
iterator.next()\n        max = maxOf(max, e)\n    }\n    return max\n}\n\n/**\n * Returns the largest element or `null` if
there are no elements.\n * \n * If any of elements is `NaN` returns `NaN`.\n * \n * The operation is _terminal_.\n
*\n@SinceKotlin("1.4")\npublic fun Sequence<Float>.maxOrNull(): Float? {\n    val iterator = iterator()\n
if (!iterator.hasNext()) return null\n    var max = iterator.next()\n    while (iterator.hasNext()) {\n        val e =
iterator.next()\n        max = maxOf(max, e)\n    }\n    return max\n}\n\n/**\n * Returns the largest element or `null` if
there are no elements.\n * \n * The operation is _terminal_.\n
*\n@SinceKotlin("1.4")\npublic fun <T :
Comparable<T>> Sequence<T>.maxOrNull(): T? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return
null\n    var max = iterator.next()\n    while (iterator.hasNext()) {\n        val e = iterator.next()\n        if (max < e) max
= e\n    }\n    return max\n}\n\n@Deprecated("Use maxWithOrNull instead.")
ReplaceWith("this.maxWithOrNull(comparator)")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\npublic fun <T> Sequence<T>.maxWith(comparator: Comparator<in T>): T? {\n
return maxWithOrNull(comparator)\n}\n\n/**\n * Returns the first element having the largest value according to the
provided [comparator] or `null` if there are no elements.\n * \n * The operation is _terminal_.\n
*\n@SinceKotlin("1.4")\npublic fun <T> Sequence<T>.maxWithOrNull(comparator: Comparator<in T>): T? {\n
val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var max = iterator.next()\n    while
(iterator.hasNext()) {\n        val e = iterator.next()\n        if (comparator.compare(max, e) < 0) max = e\n    }\n
return max\n}\n\n@Deprecated("Use minOrNull instead.")
ReplaceWith("this.minOrNull()")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\n@SinceKotlin("1.1")\npublic fun Sequence<Double>.min(): Double? {\n    return
minOrNull()\n}\n\n@Deprecated("Use minOrNull instead.")
ReplaceWith("this.minOrNull()")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\n@SinceKotlin("1.1")\npublic fun Sequence<Float>.min(): Float? {\n    return
minOrNull()\n}\n\n@Deprecated("Use minOrNull instead.")

```

```

ReplaceWith("this.minOrNull()")\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince = \"1.5\",
hiddenSince = \"1.6\")\npublic fun <T : Comparable<T>> Sequence<T>.min(): T? {\n    return
minOrNull()\n}\n\n@Deprecated("Use minByOrNull instead.")
ReplaceWith("this.minByOrNull(selector)")\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince =
\"1.5\", hiddenSince = \"1.6\")\npublic inline fun <T, R : Comparable<R>> Sequence<T>.minBy(selector: (T) ->
R): T? {\n    return minByOrNull(selector)\n}\n\n/**\n * Returns the first element yielding the smallest value of the
given function or `null` if there are no elements.\n * \n * The operation is _terminal_.\n * \n * @sample
samples.collections.Collections.Aggregates.minByOrNull\n * \n * @SinceKotlin("1.4")\n * \n * public inline fun <T, R :
Comparable<R>> Sequence<T>.minByOrNull(selector: (T) -> R): T? {\n    val iterator = iterator()\n    if
(!iterator.hasNext()) return null\n    var minElem = iterator.next()\n    if (!iterator.hasNext()) return minElem\n    var
minValue = selector(minElem)\n    do {\n        val e = iterator.next()\n        val v = selector(e)\n        if (minValue >
v) {\n            minElem = e\n            minValue = v\n        }\n    } while (iterator.hasNext())\n    return
minElem\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the sequence.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result
is `NaN`.\n * \n * @throws NoSuchElementException if the sequence is empty.\n * \n * The operation is
_terminal_.\n
*\n * @SinceKotlin("1.4")\n * \n * @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n * \n * @OverloadResolution
ByLambdaReturnType\n * \n * @kotlin.internal.InlineOnly\n * \n * public inline fun <T> Sequence<T>.minOf(selector: (T) ->
Double): Double {\n    val iterator = iterator()\n    if (!iterator.hasNext()) throw NoSuchElementException()\n    var
minValue = selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v = selector(iterator.next())\n
minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all
values produced by [selector] function\n * applied to each element in the sequence.\n * \n * If any of values
produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException
if the sequence is empty.\n * \n * The operation is _terminal_.\n
*\n * @SinceKotlin("1.4")\n * \n * @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n * \n * @OverloadResolution
ByLambdaReturnType\n * \n * @kotlin.internal.InlineOnly\n * \n * public inline fun <T> Sequence<T>.minOf(selector: (T) ->
Float): Float {\n    val iterator = iterator()\n    if (!iterator.hasNext()) throw NoSuchElementException()\n    var
minValue = selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v = selector(iterator.next())\n
minValue = minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all
values produced by [selector] function\n * applied to each element in the sequence.\n * \n * @throws
NoSuchElementException if the sequence is empty.\n * \n * The operation is _terminal_.\n
*\n * @SinceKotlin("1.4")\n * \n * @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n * \n * @OverloadResolution
ByLambdaReturnType\n * \n * @kotlin.internal.InlineOnly\n * \n * public inline fun <T, R : Comparable<R>>
Sequence<T>.minOf(selector: (T) -> R): R {\n    val iterator = iterator()\n    if (!iterator.hasNext()) throw
NoSuchElementException()\n    var minValue = selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v
= selector(iterator.next())\n        if (minValue > v) {\n            minValue = v\n        }\n    }\n    return
minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to
each element in the sequence or `null` if there are no elements.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n * \n * The operation is _terminal_.\n
*\n * @SinceKotlin("1.4")\n * \n * @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n * \n * @OverloadResolution
ByLambdaReturnType\n * \n * @kotlin.internal.InlineOnly\n * \n * public inline fun <T> Sequence<T>.minOrNull(selector:
(T) -> Double): Double? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var minValue =
selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v = selector(iterator.next())\n        minValue =
minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the sequence or `null` if there are no elements.\n * \n * If any of
values produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * The operation is _terminal_.\n
*\n * @SinceKotlin("1.4")\n * \n * @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n * \n * @OverloadResolution
ByLambdaReturnType\n * \n * @kotlin.internal.InlineOnly\n * \n * public inline fun <T> Sequence<T>.minOrNull(selector:

```

```

(T) -> Float): Float? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var minValue =
selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v = selector(iterator.next())\n        minValue =
minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the sequence or `null` if there are no elements.\n *\n * The
operation is _terminal_.\n
*\n*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R : Comparable<R>>
Sequence<T>.minOfOrNull(selector: (T) -> R): R? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return
null\n    var minValue = selector(iterator.next())\n    while (iterator.hasNext()) {\n        val v =
selector(iterator.next())\n        if (minValue > v) {\n            minValue = v\n        }\n    }\n    return
minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values
produced by [selector] function applied to each element in the sequence.\n *\n * @throws
NoSuchElementException if the sequence is empty.\n *\n * The operation is _terminal_.\n
*\n*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R>
Sequence<T>.minOfWith(comparator: Comparator<in R>, selector: (T) -> R): R {\n    val iterator = iterator()\n    if
(!iterator.hasNext()) throw NoSuchElementException()\n    var minValue = selector(iterator.next())\n    while
(iterator.hasNext()) {\n        val v = selector(iterator.next())\n        if (comparator.compare(minValue, v) > 0) {\n
            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value according to the
provided [comparator]\n * among all values produced by [selector] function applied to each element in the sequence
or `null` if there are no elements.\n *\n * The operation is _terminal_.\n
*\n*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <T, R>
Sequence<T>.minOfWithOrNull(comparator: Comparator<in R>, selector: (T) -> R): R? {\n    val iterator =
iterator()\n    if (!iterator.hasNext()) return null\n    var minValue = selector(iterator.next())\n    while
(iterator.hasNext()) {\n        val v = selector(iterator.next())\n        if (comparator.compare(minValue, v) > 0) {\n
            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest element or `null` if there are
no elements.\n *\n * If any of elements is `NaN` returns `NaN`.\n *\n * The operation is _terminal_.\n
*\n*\n@SinceKotlin("1.4")\npublic fun Sequence<Double>.minOrNull(): Double? {\n    val iterator = iterator()\n    if
(!iterator.hasNext()) return null\n    var min = iterator.next()\n    while (iterator.hasNext()) {\n        val e =
iterator.next()\n        min = minOf(min, e)\n    }\n    return min\n}\n\n/**\n * Returns the smallest element or `null` if
there are no elements.\n *\n * If any of elements is `NaN` returns `NaN`.\n *\n * The operation is _terminal_.\n
*\n*\n@SinceKotlin("1.4")\npublic fun Sequence<Float>.minOrNull(): Float? {\n    val iterator = iterator()\n    if
(!iterator.hasNext()) return null\n    var min = iterator.next()\n    while (iterator.hasNext()) {\n        val e =
iterator.next()\n        min = minOf(min, e)\n    }\n    return min\n}\n\n/**\n * Returns the smallest element or `null` if
there are no elements.\n *\n * The operation is _terminal_.\n
*\n*\n@SinceKotlin("1.4")\npublic fun <T :
Comparable<T>> Sequence<T>.minOrNull(): T? {\n    val iterator = iterator()\n    if (!iterator.hasNext()) return
null\n    var min = iterator.next()\n    while (iterator.hasNext()) {\n        val e = iterator.next()\n        if (min > e) min
= e\n    }\n    return min\n}\n\n@Deprecated("Use minWithOrNull instead.")
ReplaceWith("this.minWithOrNull(comparator)")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince
= "1.5", hiddenSince = "1.6")\npublic fun <T> Sequence<T>.minWith(comparator: Comparator<in T>): T? {\n
return minWithOrNull(comparator)\n}\n\n/**\n * Returns the first element having the smallest value according to
the provided [comparator] or `null` if there are no elements.\n *\n * The operation is _terminal_.\n
*\n*\n@SinceKotlin("1.4")\npublic fun <T> Sequence<T>.minWithOrNull(comparator: Comparator<in T>): T? {\n
val iterator = iterator()\n    if (!iterator.hasNext()) return null\n    var min = iterator.next()\n    while
(iterator.hasNext()) {\n        val e = iterator.next()\n        if (comparator.compare(min, e) > 0) min = e\n    }\n    return
min\n}\n\n/**\n * Returns `true` if the sequence has no elements.\n *\n * The operation is _terminal_.\n
*\n *\n * @sample samples.collections.Collections.Aggregates.none\n *\n*\n@SinceKotlin("1.4")\npublic fun <T> Sequence<T>.none(): Boolean {\n

```

```

return iterator().hasNext()\n\n/**\n * Returns `true` if no elements match the given [predicate].\n *\n * The operation is _terminal_.\n *\n * @sample samples.collections.Collections.Aggregates.noneWithPredicate\n *\npublic inline fun <T> Sequence<T>.none(predicate: (T) -> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return false\n    return true\n}\n\n/**\n * Returns a sequence which performs the given [action] on each element of the original sequence as they pass through it.\n *\n * The operation is _intermediate_ and _stateless_.\n *\n@SinceKotlin("1.1")\npublic fun <T> Sequence<T>.onEach(action: (T) -> Unit): Sequence<T> {\n    return map {\n        action(it)\n        it\n    }\n}\n\n/**\n * Returns a sequence which performs the given [action] on each element of the original sequence as they pass through it.\n *\n * @param [action] function that takes the index of an element and the element itself\n * and performs the action on the element.\n *\n * The operation is _intermediate_ and _stateless_.\n *\n@SinceKotlin("1.4")\npublic fun <T> Sequence<T>.onEachIndexed(action: (index: Int, T) -> Unit): Sequence<T> {\n    return mapIndexed { index, element ->\n        action(index, element)\n        element\n    }\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from left to right\n * to current accumulator value and each element.\n *\n * Throws an exception if this sequence is empty. If the sequence can be empty in an expected way,\n * please use [reduceOrNull] instead. It returns `null` when its receiver is empty.\n *\n * @param [operation] function that takes current accumulator value and an element,\n * and calculates the next accumulator value.\n *\n * The operation is _terminal_.\n *\n * @sample samples.collections.Collections.Aggregates.reduce\n *\npublic inline fun <S, T : S> Sequence<T>.reduce(operation: (acc: S, T) -> S): S {\n    val iterator = this.iterator()\n    if (!iterator.hasNext()) throw UnsupportedOperationException("Empty sequence can't be reduced.")\n    var accumulator: S = iterator.next()\n    while (iterator.hasNext()) {\n        accumulator = operation(accumulator, iterator.next())\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from left to right\n * to current accumulator value and each element with its index in the original sequence.\n *\n * Throws an exception if this sequence is empty. If the sequence can be empty in an expected way,\n * please use [reduceIndexedOrNull] instead. It returns `null` when its receiver is empty.\n *\n * @param [operation] function that takes the index of an element, current accumulator value and the element itself,\n * and calculates the next accumulator value.\n *\n * The operation is _terminal_.\n *\n * @sample samples.collections.Collections.Aggregates.reduce\n *\npublic inline fun <S, T : S> Sequence<T>.reduceIndexed(operation: (index: Int, acc: S, T) -> S): S {\n    val iterator = this.iterator()\n    if (!iterator.hasNext()) throw UnsupportedOperationException("Empty sequence can't be reduced.")\n    var index = 1\n    var accumulator: S = iterator.next()\n    while (iterator.hasNext()) {\n        accumulator = operation(checkIndexOverflow(index++), accumulator, iterator.next())\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from left to right\n * to current accumulator value and each element with its index in the original sequence.\n *\n * Returns `null` if the sequence is empty.\n *\n * @param [operation] function that takes the index of an element, current accumulator value and the element itself,\n * and calculates the next accumulator value.\n *\n * The operation is _terminal_.\n *\n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n *\n@SinceKotlin("1.4")\npublic inline fun <S, T : S> Sequence<T>.reduceIndexedOrNull(operation: (index: Int, acc: S, T) -> S): S? {\n    val iterator = this.iterator()\n    if (!iterator.hasNext()) return null\n    var index = 1\n    var accumulator: S = iterator.next()\n    while (iterator.hasNext()) {\n        accumulator = operation(checkIndexOverflow(index++), accumulator, iterator.next())\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting with the first element and applying [operation] from left to right\n * to current accumulator value and each element.\n *\n * Returns `null` if the sequence is empty.\n *\n * @param [operation] function that takes current accumulator value and an element,\n * and calculates the next accumulator value.\n *\n * The operation is _terminal_.\n *\n * @sample samples.collections.Collections.Aggregates.reduceOrNull\n *\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun <S, T : S> Sequence<T>.reduceOrNull(operation: (acc: S, T) -> S): S? {\n    val iterator = this.iterator()\n    if (!iterator.hasNext()) return null\n    var accumulator: S = iterator.next()\n    while (iterator.hasNext()) {\n        accumulator = operation(accumulator, iterator.next())\n    }\n    return accumulator\n}\n\n/**\n * Returns a sequence

```

containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with [initial] value. Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting sequence. The [initial] value should also be immutable (or should not be mutated) as it may be passed to [operation] function later because of sequence's lazy nature.

@param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value. The operation is `_intermediate_` and `_stateless_`.

@sample samples.collections.Collections.Aggregates.runningFold

```

*^@SinceKotlin("1.4")
public fun <T, R> Sequence<T>.runningFold(initial: R, operation: (acc: R, T) -> R):
Sequence<R> {
    return sequence {
        yield(initial)
        var accumulator = initial
        for (element in
this@runningFold) {
            accumulator = operation(accumulator, element)
            yield(accumulator)
        }
    }
}

```

Returns a sequence containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original sequence and current accumulator value that starts with [initial] value. Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting sequence. The [initial] value should also be immutable (or should not be mutated) as it may be passed to [operation] function later because of sequence's lazy nature.

@param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value. The operation is `_intermediate_` and `_stateless_`.

@sample samples.collections.Collections.Aggregates.runningFoldIndexed

```

*^@SinceKotlin("1.4")
public fun
<T, R> Sequence<T>.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, T) -> R): Sequence<R> {
    return sequence {
        yield(initial)
        var index = 0
        var accumulator = initial
        for (element in
this@runningFoldIndexed) {
            accumulator = operation(checkIndexOverflow(index++), accumulator,
element)
            yield(accumulator)
        }
    }
}

```

Returns a sequence containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with the first element of this sequence. Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting sequence.

@param [operation] function that takes current accumulator value and the element, and calculates the next accumulator value. The operation is `_intermediate_` and `_stateless_`.

@sample samples.collections.Collections.Aggregates.runningReduce

```

*^@SinceKotlin("1.4")
@WasExperimental(ExperimentalStdlibApi::class)
public fun <S, T : S>
Sequence<T>.runningReduce(operation: (acc: S, T) -> S): Sequence<S> {
    return sequence {
        val iterator =
iterator()
        if (iterator.hasNext()) {
            var accumulator: S = iterator.next()
            yield(accumulator)
            while (iterator.hasNext()) {
                accumulator = operation(accumulator, iterator.next())
            }
        }
    }
}

```

Returns a sequence containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original sequence and current accumulator value that starts with the first element of this sequence. Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting sequence.

@param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value. The operation is `_intermediate_` and `_stateless_`.

@sample samples.collections.Collections.Aggregates.runningReduceIndexed

```

*^@SinceKotlin("1.4")
public fun
<S, T : S> Sequence<T>.runningReduceIndexed(operation: (index: Int, acc: S, T) -> S): Sequence<S> {
    return
sequence {
        val iterator = iterator()
        if (iterator.hasNext()) {
            var accumulator: S =
iterator.next()
            yield(accumulator)
            var index = 1
            while (iterator.hasNext()) {
                accumulator = operation(checkIndexOverflow(index++), accumulator, iterator.next())
            }
        }
    }
}

```

Returns a sequence containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with [initial] value. Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting sequence. The [initial] value should also be immutable (or should not be mutated) as it may be passed to [operation] function later because of sequence's lazy nature.

```

* \n * @param [operation] function that takes current accumulator value and an element, and calculates the next
accumulator value.\n * \n * The operation is _intermediate_ and _stateless_.\n * \n * @sample
samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun <T, R>
Sequence<T>.scan(initial: R, operation: (acc: R, T) -> R): Sequence<R> {\n    return runningFold(initial,
operation)\n}\n\n/**\n * Returns a sequence containing successive accumulation values generated by applying
[operation] from left to right\n * to each element, its index in the original sequence and current accumulator value
that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n
* otherwise it would affect the previous value in resulting sequence.\n * The [initial] value should also be immutable
(or should not be mutated)\n * as it may be passed to [operation] function later because of sequence's lazy nature.\n
* \n * @param [operation] function that takes the index of an element, current accumulator value\n * and the
element itself, and calculates the next accumulator value.\n * \n * The operation is _intermediate_ and _stateless_.\n
* \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun <T, R>
Sequence<T>.scanIndexed(initial: R, operation: (index: Int, acc: R, T) -> R): Sequence<R> {\n    return
runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the sequence.\n * \n * The operation is _terminal_.\n *\n@Deprecated("Use sumOf
instead.", ReplaceWith("this.sumOf(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline
fun <T> Sequence<T>.sumBy(selector: (T) -> Int): Int {\n    var sum: Int = 0\n    for (element in this) {\n        sum
+= selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the sequence.\n * \n * The operation is _terminal_.\n *\n@Deprecated("Use
sumOf instead.", ReplaceWith("this.sumOf(selector)"))\n@DeprecatedSinceKotlin(warningSince =
"1.5")\npublic inline fun <T> Sequence<T>.sumByDouble(selector: (T) -> Double): Double {\n    var sum: Double
= 0.0\n    for (element in this) {\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum
of all values produced by [selector] function applied to each element in the sequence.\n * \n * The operation is
_terminal_.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfDouble")\n@kotlin.internal.InlineOnly\npublic inline fun
<T> Sequence<T>.sumOf(selector: (T) -> Double): Double {\n    var sum: Double = 0.toDouble()\n    for (element
in this) {\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced
by [selector] function applied to each element in the sequence.\n * \n * The operation is _terminal_.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfInt")\n@kotlin.internal.InlineOnly\npublic inline fun <T>
Sequence<T>.sumOf(selector: (T) -> Int): Int {\n    var sum: Int = 0.toInt()\n    for (element in this) {\n        sum
+= selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the sequence.\n * \n * The operation is _terminal_.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfLong")\n@kotlin.internal.InlineOnly\npublic inline fun
<T> Sequence<T>.sumOf(selector: (T) -> Long): Long {\n    var sum: Long = 0.toLong()\n    for (element in this)
{\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each element in the sequence.\n * \n * The operation is _terminal_.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfUInt")\n@WasExperimental(ExperimentalUnsignedType
s::class)\n@kotlin.internal.InlineOnly\npublic inline fun <T> Sequence<T>.sumOf(selector: (T) -> UInt): UInt {\n
    var sum: UInt = 0.toUInt()\n    for (element in this) {\n        sum += selector(element)\n    }\n    return
sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the
sequence.\n * \n * The operation is _terminal_.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution

```

```

ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedTy
pes::class)\n@kotlin.internal.InlineOnly\npublic inline fun <T> Sequence<T>.sumOf(selector: (T) -> ULong):
ULong {\n    var sum: ULong = 0.toULong()\n    for (element in this) {\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns an original collection containing all the non-`null` elements, throwing an
[IllegalArgumentException] if there are any `null` elements.\n * \n * The operation is _intermediate_ and
_stateless_.\n * \n * @public fun <T : Any> Sequence<T?>.requireNonNulls(): Sequence<T> {\n    return map { it ?
: throw IllegalArgumentException("null element found in $this.") }\n}\n\n/**\n * Splits this sequence into a
sequence of lists each not exceeding the given [size].\n * \n * The last list in the resulting sequence may have fewer
elements than the given [size].\n * \n * @param size the number of elements to take in each list, must be positive
and can be greater than the number of elements in this sequence.\n * \n * The operation is _intermediate_ and
_stateful_.\n * \n * @sample samples.collections.Collections.Transformations.chunked\n
*\n*\n@SinceKotlin("1.2")\npublic fun <T> Sequence<T>.chunked(size: Int): Sequence<List<T>> {\n    return
windowed(size, size, partialWindows = true)\n}\n\n/**\n * Splits this sequence into several lists each not exceeding
the given [size]\n * and applies the given [transform] function to an each.\n * \n * @return sequence of results of the
[transform] applied to an each list.\n * \n * Note that the list passed to the [transform] function is ephemeral and is
valid only inside that function.\n * You should not store it or allow it to escape in some way, unless you made a
snapshot of it.\n * The last list may have fewer elements than the given [size].\n * \n * @param size the number of
elements to take in each list, must be positive and can be greater than the number of elements in this sequence.\n * \n
*\n * The operation is _intermediate_ and _stateful_.\n * \n * @sample samples.text.Strings.chunkedTransform\n
*\n*\n@SinceKotlin("1.2")\npublic fun <T, R> Sequence<T>.chunked(size: Int, transform: (List<T>) -> R):
Sequence<R> {\n    return windowed(size, size, partialWindows = true, transform = transform)\n}\n\n/**\n *
Returns a sequence containing all elements of the original sequence without the first occurrence of the given
[element].\n * \n * The operation is _intermediate_ and _stateless_.\n * \n * @public operator fun <T>
Sequence<T>.minus(element: T): Sequence<T> {\n    return object: Sequence<T> {\n        override fun iterator():
Iterator<T> {\n            var removed = false\n            return this@minus.filter { if (!removed && it == element) {\n
                removed = true; false } else true }.iterator()\n        }\n    }\n}\n\n/**\n * Returns a sequence containing all elements
of original sequence except the elements contained in the given [elements] array.\n * \n * Note that the source
sequence and the array being subtracted are iterated only when an `iterator` is requested from\n * the resulting
sequence. Changing any of them between successive calls to `iterator` may affect the result.\n * \n * Before Kotlin
1.6, the [elements] array may have been converted to a [HashSet] to speed up the operation, thus the elements were
required to have\n * a correct and stable implementation of `hashCode()` that didn't change between successive
invocations.\n * On JVM, you can enable this behavior back with the system property
`kotlin.collections.convert_arg_to_set_in_removeAll` set to `true`.\n * \n * The operation is _intermediate_ and
_stateful_.\n * \n * @public operator fun <T> Sequence<T>.minus(elements: Array<out T>): Sequence<T> {\n    if
(elements.isEmpty()) return this\n    return object: Sequence<T> {\n        override fun iterator(): Iterator<T> {\n
            val other = elements.convertToSetForSetOperation()\n            return this@minus.filterNot { it in other }.iterator()\n
        }\n    }\n}\n\n/**\n * Returns a sequence containing all elements of original sequence except the elements
contained in the given [elements] collection.\n * \n * Note that the source sequence and the collection being
subtracted are iterated only when an `iterator` is requested from\n * the resulting sequence. Changing any of them
between successive calls to `iterator` may affect the result.\n * \n * Before Kotlin 1.6, the [elements] collection may
have been converted to a [HashSet] to speed up the operation, thus the elements were required to have\n * a correct
and stable implementation of `hashCode()` that didn't change between successive invocations.\n * On JVM, you can
enable this behavior back with the system property `kotlin.collections.convert_arg_to_set_in_removeAll` set to
`true`.\n * \n * The operation is _intermediate_ and _stateful_.\n * \n * @public operator fun <T>
Sequence<T>.minus(elements: Iterable<T>): Sequence<T> {\n    return object: Sequence<T> {\n        override fun
iterator(): Iterator<T> {\n            val other = elements.convertToSetForSetOperation()\n            if (other.isEmpty())\n
                return this@minus.iterator()\n            else\n                return this@minus.filterNot { it in other }.iterator()\n
        }\n    }\n}\n\n/**\n * Returns a sequence containing all elements of original sequence except the elements

```

contained in the given [elements] sequence. Note that the source sequence and the sequence being subtracted are iterated only when an `iterator` is requested from the resulting sequence. Changing any of them between successive calls to `iterator` may affect the result. The operation is `_intermediate_` for this sequence and `_terminal_` and `_stateful_` for the [elements] sequence. Before Kotlin 1.6, the [elements] sequence may have been converted to a [HashSet] to speed up the operation, thus the elements were required to have a correct and stable implementation of `hashCode()` that didn't change between successive invocations. On JVM, you can enable this behavior back with the system property `kotlin.collections.convert_arg_to_set_in_removeAll` set to `true`.

```

public operator fun <T> Sequence<T>.minus(elements: Sequence<T>): Sequence<T> {
    return object: Sequence<T> {
        override fun iterator(): Iterator<T> {
            val other = elements.convertToSetForSetOperation()
            if (other.isEmpty()) return this@minus.iterator()
            else return this@minus.filterNot { it in other }.iterator()
        }
    }
}

```

\* Returns a sequence containing all elements of the original sequence without the first occurrence of the given [element]. The operation is `_intermediate_` and `_stateless_`.

```

@kotlin.internal.InlineOnly
public inline fun <T> Sequence<T>.minusElement(element: T): Sequence<T> {
    return minus(element)
}

```

\* Splits the original sequence into pair of lists, where `*first*` list contains elements for which [predicate] yielded `true`, while `*second*` list contains elements for which [predicate] yielded `false`. The operation is `_terminal_`.

```

@sample.samples.collections.Sequences.Transformations.partition
public inline fun <T> Sequence<T>.partition(predicate: (T) -> Boolean): Pair<List<T>, List<T>> {
    val first = ArrayList<T>()
    val second = ArrayList<T>()
    for (element in this) {
        if (predicate(element)) first.add(element)
    } else {
        second.add(element)
    }
    return Pair(first, second)
}

```

\* Returns a sequence containing all elements of the original sequence and then the given [element]. The operation is `_intermediate_` and `_stateless_`.

```

public operator fun <T> Sequence<T>.plus(element: T): Sequence<T> {
    return sequenceOf(this, sequenceOf(element)).flatten()
}

```

\* Returns a sequence containing all elements of original sequence and then all elements of the given [elements] array. Note that the source sequence and the array being added are iterated only when an `iterator` is requested from the resulting sequence. Changing any of them between successive calls to `iterator` may affect the result. The operation is `_intermediate_` and `_stateless_`.

```

public operator fun <T> Sequence<T>.plus(elements: Array<out T>): Sequence<T> {
    return this.plus(elements.asList())
}

```

\* Returns a sequence containing all elements of original sequence and then all elements of the given [elements] collection. Note that the source sequence and the collection being added are iterated only when an `iterator` is requested from the resulting sequence. Changing any of them between successive calls to `iterator` may affect the result. The operation is `_intermediate_` and `_stateless_`.

```

public operator fun <T> Sequence<T>.plus(elements: Iterable<T>): Sequence<T> {
    return sequenceOf(this, elements.asSequence()).flatten()
}

```

\* Returns a sequence containing all elements of original sequence and then all elements of the given [elements] sequence. Note that the source sequence and the sequence being added are iterated only when an `iterator` is requested from the resulting sequence. Changing any of them between successive calls to `iterator` may affect the result. The operation is `_intermediate_` and `_stateless_`.

```

public operator fun <T> Sequence<T>.plus(elements: Sequence<T>): Sequence<T> {
    return sequenceOf(this, elements).flatten()
}

```

\* Returns a sequence containing all elements of the original sequence and then the given [element]. The operation is `_intermediate_` and `_stateless_`.

```

@kotlin.internal.InlineOnly
public inline fun <T> Sequence<T>.plusElement(element: T): Sequence<T> {
    return plus(element)
}

```

\* Returns a sequence of snapshots of the window of the given [size] sliding along this sequence with the given [step], where each snapshot is a list. Several last lists may have fewer elements than the given [size]. Both [size] and [step] must be positive and can be greater than the number of elements in this sequence. @param size the number of elements to take in each window @param step the number of elements to move the window forward by on an each step, by default 1 @param partialWindows controls whether or not to keep partial windows in the end if any, by default `false` which means partial windows won't be preserved

```

@sample.samples.collections.Sequences.Transformations.takeWindows
@SinceKotlin("1.2")
public fun <T> Sequence<T>.windowed(size: Int, step: Int = 1, partialWindows:

```

```

Boolean = false): Sequence<List<T>> {\n  return windowedSequence(size, step, partialWindows, reuseBuffer =
false)\n}\n\n/**\n * Returns a sequence of results of applying the given [transform] function to\n * an each list
representing a view over the window of the given [size]\n * sliding along this sequence with the given [step].\n * \n
* Note that the list passed to the [transform] function is ephemeral and is valid only inside that function.\n * You
should not store it or allow it to escape in some way, unless you made a snapshot of it.\n * Several last lists may
have fewer elements than the given [size].\n * \n * Both [size] and [step] must be positive and can be greater than
the number of elements in this sequence.\n * @param size the number of elements to take in each window\n *
@param step the number of elements to move the window forward by on an each step, by default 1\n * @param
partialWindows controls whether or not to keep partial windows in the end if any,\n * by default `false` which
means partial windows won't be preserved\n * \n * @sample
samples.collections.Sequences.Transformations.averageWindows\n *\n@SinceKotlin("1.2")\npublic fun <T, R>
Sequence<T>.windowed(size: Int, step: Int = 1, partialWindows: Boolean = false, transform: (List<T>) -> R):
Sequence<R> {\n  return windowedSequence(size, step, partialWindows, reuseBuffer =
true).map(transform)\n}\n\n/**\n * Returns a sequence of values built from the elements of `this` sequence and the
[other] sequence with the same index.\n * The resulting sequence ends as soon as the shortest input sequence ends.\n
*\n * The operation is _intermediate_ and _stateless_.\n * \n * @sample
samples.collections.Sequences.Transformations.zip\n *\n@SinceKotlin("1.2")\npublic infix fun <T, R> Sequence<T>.zip(other:
Sequence<R>): Sequence<Pair<T, R>> {\n  return MergingSequence(this, other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n *
Returns a sequence of values built from the elements of `this` sequence and the [other] sequence with the same
index\n * using the provided [transform] function applied to each pair of elements.\n * The resulting sequence ends
as soon as the shortest input sequence ends.\n * \n * The operation is _intermediate_ and _stateless_.\n * \n *
@param sample samples.collections.Sequences.Transformations.zipWithTransform\n *\n@SinceKotlin("1.2")\npublic fun <T, R, V>
Sequence<T>.zip(other: Sequence<R>, transform: (a: T, b: R) -> V): Sequence<V> {\n  return
MergingSequence(this, other, transform)\n}\n\n/**\n * Returns a sequence of pairs of each two adjacent elements in
this sequence.\n * \n * The returned sequence is empty if this sequence contains less than two elements.\n * \n * The
operation is _intermediate_ and _stateless_.\n * \n * @sample
samples.collections.Collections.Transformations.zipWithNext\n *\n@SinceKotlin("1.2")\npublic fun <T>
Sequence<T>.zipWithNext(): Sequence<Pair<T, T>> {\n  return zipWithNext { a, b -> a to b }\n}\n\n/**\n *
Returns a sequence containing the results of applying the given [transform] function\n * to an each pair of two
adjacent elements in this sequence.\n * \n * The returned sequence is empty if this sequence contains less than two
elements.\n * \n * The operation is _intermediate_ and _stateless_.\n * \n * @sample
samples.collections.Collections.Transformations.zipWithNextToFindDeltas\n *\n@SinceKotlin("1.2")\npublic
fun <T, R> Sequence<T>.zipWithNext(transform: (a: T, b: T) -> R): Sequence<R> {\n  return sequence result@
{\n    val iterator = iterator()\n    if (!iterator.hasNext()) return@result\n    var current = iterator.next()\n
while (iterator.hasNext()) {\n    val next = iterator.next()\n    yield(transform(current, next))\n
current = next\n    }\n  }\n}\n\n/**\n * Appends the string from all the elements separated using [separator] and
using the given [prefix] and [postfix] if supplied.\n * \n * If the collection could be huge, you can specify a non-
negative value of [limit], in which case only the first [limit]\n * elements will be appended, followed by the
[truncated] string (which defaults to "...").\n * \n * The operation is _terminal_.\n * \n * @sample
samples.collections.Collections.Transformations.joinTo\n *\n@SinceKotlin("1.2")\npublic fun <T, A : Appendable>
Sequence<T>.joinTo(buffer: A, separator: CharSequence = "\", \"", prefix: CharSequence = "\"", postfix:
CharSequence = "\"", limit: Int = -1, truncated: CharSequence = "...\", transform: ((T) -> CharSequence)? = null): A
{\n  buffer.append(prefix)\n  var count = 0\n  for (element in this) {\n    if (++count > 1)
buffer.append(separator)\n    if (limit < 0 || count <= limit) {\n      buffer.appendElement(element, transform)\n
    } else break\n  }\n  if (limit >= 0 && count > limit) buffer.append(truncated)\n  buffer.append(postfix)\n
return buffer\n}\n\n/**\n * Creates a string from all the elements separated using [separator] and using the given
[prefix] and [postfix] if supplied.\n * \n * If the collection could be huge, you can specify a non-negative value of
[limit], in which case only the first [limit]\n * elements will be appended, followed by the [truncated] string (which

```

```

defaults to "...").\n * \n * The operation is _terminal_.\n * \n * @sample
samples.collections.Collections.Transformations.joinToString\n * \npublic fun <T>
Sequence<T>.joinToString(separator: CharSequence = "\", \"", prefix: CharSequence = "\"", postfix: CharSequence =
\"", limit: Int = -1, truncated: CharSequence = "...\", transform: ((T) -> CharSequence)? = null): String {\n    return
joinTo(StringBuilder(), separator, prefix, postfix, limit, truncated, transform).toString()\n}\n\n/**\n * Creates an
[Iterable] instance that wraps the original sequence returning its elements when being iterated.\n * \npublic fun <T>
Sequence<T>.asIterable(): Iterable<T> {\n    return Iterable { this.iterator() }\n}\n\n/**\n * Returns this sequence as
a [Sequence].\n * \n@kotlin.internal.InlineOnly\npublic inline fun <T> Sequence<T>.asSequence(): Sequence<T>
{\n    return this\n}\n\n/**\n * Returns an average value of elements in the sequence.\n * \n * The operation is
_terminal_.\n * \n@kotlin.jvm.JvmName("averageOfByte")\npublic fun Sequence<Byte>.average(): Double {\n    var sum:
Double = 0.0\n    var count: Int = 0\n    for (element in this) {\n        sum += element\n        checkCountOverflow(++count)\n    }\n    return if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns
an average value of elements in the sequence.\n * \n * The operation is _terminal_.\n * \n@kotlin.jvm.JvmName("averageOfShort")\npublic fun Sequence<Short>.average(): Double {\n    var sum:
Double = 0.0\n    var count: Int = 0\n    for (element in this) {\n        sum += element\n        checkCountOverflow(++count)\n    }\n    return if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns
an average value of elements in the sequence.\n * \n * The operation is _terminal_.\n * \n@kotlin.jvm.JvmName("averageOfInt")\npublic fun Sequence<Int>.average(): Double {\n    var sum: Double
= 0.0\n    var count: Int = 0\n    for (element in this) {\n        sum += element\n        checkCountOverflow(++count)\n    }\n    return if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns an average value of elements in the
sequence.\n * \n * The operation is _terminal_.\n * \n@kotlin.jvm.JvmName("averageOfLong")\npublic fun
Sequence<Long>.average(): Double {\n    var sum: Double = 0.0\n    var count: Int = 0\n    for (element in this) {\n
        sum += element\n        checkCountOverflow(++count)\n    }\n    return if (count == 0) Double.NaN else sum /
count\n}\n\n/**\n * Returns an average value of elements in the sequence.\n * \n * The operation is _terminal_.\n * \n@kotlin.jvm.JvmName("averageOfFloat")\npublic fun Sequence<Float>.average(): Double {\n    var sum:
Double = 0.0\n    var count: Int = 0\n    for (element in this) {\n        sum += element\n        checkCountOverflow(++count)\n    }\n    return if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns
an average value of elements in the sequence.\n * \n * The operation is _terminal_.\n * \n@kotlin.jvm.JvmName("averageOfDouble")\npublic fun Sequence<Double>.average(): Double {\n    var sum:
Double = 0.0\n    var count: Int = 0\n    for (element in this) {\n        sum += element\n        checkCountOverflow(++count)\n    }\n    return if (count == 0) Double.NaN else sum / count\n}\n\n/**\n * Returns
the sum of all elements in the sequence.\n * \n * The operation is _terminal_.\n * \n@kotlin.jvm.JvmName("sumOfByte")\npublic fun Sequence<Byte>.sum(): Int {\n    var sum: Int = 0\n    for
(element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in the
sequence.\n * \n * The operation is _terminal_.\n * \n@kotlin.jvm.JvmName("sumOfShort")\npublic fun
Sequence<Short>.sum(): Int {\n    var sum: Int = 0\n    for (element in this) {\n        sum += element\n    }\n    return
sum\n}\n\n/**\n * Returns the sum of all elements in the sequence.\n * \n * The operation is _terminal_.\n * \n@kotlin.jvm.JvmName("sumOfInt")\npublic fun Sequence<Int>.sum(): Int {\n    var sum: Int = 0\n    for
(element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in the
sequence.\n * \n * The operation is _terminal_.\n * \n@kotlin.jvm.JvmName("sumOfLong")\npublic fun
Sequence<Long>.sum(): Long {\n    var sum: Long = 0L\n    for (element in this) {\n        sum += element\n    }\n    return
sum\n}\n\n/**\n * Returns the sum of all elements in the sequence.\n * \n * The operation is _terminal_.\n * \n@kotlin.jvm.JvmName("sumOfFloat")\npublic fun Sequence<Float>.sum(): Float {\n    var sum: Float = 0.0f\n    for
(element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in
the sequence.\n * \n * The operation is _terminal_.\n * \n@kotlin.jvm.JvmName("sumOfDouble")\npublic fun
Sequence<Double>.sum(): Double {\n    var sum: Double = 0.0\n    for (element in this) {\n        sum += element\n    }\n    return
sum\n}\n\n"/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the

```

license/LICENSE.txt file.\n

```
*\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName(\"SetsKt\")\n\npackage  
kotlin.collections\n\n// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt\n// See:  
https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\nimport kotlin.random.*\nimport  
kotlin.ranges.contains\nimport kotlin.ranges.reversed\n\n/**\n * Returns a set containing all elements of the original  
set except the given [element].\n */\n * The returned set preserves the element iteration order of the original set.\n\n * \n\npublic operator fun <T> Set<T>.minus(element: T): Set<T> {\n    val result =  
LinkedHashSet<T>(mapCapacity(size))\n    var removed = false\n    return this.filterTo(result) { if (!removed && it  
== element) { removed = true; false } else true }\n}\n\n/**\n * Returns a set containing all elements of the original  
set except the elements contained in the given [elements] array.\n */\n * The returned set preserves the element  
iteration order of the original set.\n * \n * Before Kotlin 1.6, the [elements] array may have been converted to a  
[HashSet] to speed up the operation, thus the elements were required to have  
 * a correct and stable implementation  
of `hashCode()` that didn't change between successive invocations.\n * On JVM, you can enable this behavior back  
with the system property `kotlin.collections.convert_arg_to_set_in_removeAll` set to `true`.\n * \n\npublic operator  
fun <T> Set<T>.minus(elements: Array<out T>): Set<T> {\n    val result = LinkedHashSet<T>(this)\n    result.removeAll(elements)\n    return result\n}\n\n/**\n * Returns a set containing all elements of the original set  
except the elements contained in the given [elements] collection.\n */\n * The returned set preserves the element  
iteration order of the original set.\n * \n * Before Kotlin 1.6, the [elements] collection may have been converted to a  
[HashSet] to speed up the operation, thus the elements were required to have  
 * a correct and stable implementation  
of `hashCode()` that didn't change between successive invocations.\n * On JVM, you can enable this behavior back  
with the system property `kotlin.collections.convert_arg_to_set_in_removeAll` set to `true`.\n * \n\npublic operator  
fun <T> Set<T>.minus(elements: Iterable<T>): Set<T> {\n    val other =  
elements.convertToSetForSetOperationWith(this)\n    if (other.isEmpty())\n        return this.toSet()\n    if (other is  
Set)\n        return this.filterNotTo(LinkedHashSet<T>()) { it in other }\n    val result = LinkedHashSet<T>(this)\n    result.removeAll(other)\n    return result\n}\n\n/**\n * Returns a set containing all elements of the original set  
except the elements contained in the given [elements] sequence.\n */\n * The returned set preserves the element  
iteration order of the original set.\n * \n * Before Kotlin 1.6, the [elements] sequence may have been converted to a  
[HashSet] to speed up the operation, thus the elements were required to have  
 * a correct and stable implementation  
of `hashCode()` that didn't change between successive invocations.\n * On JVM, you can enable this behavior back  
with the system property `kotlin.collections.convert_arg_to_set_in_removeAll` set to `true`.\n * \n\npublic operator  
fun <T> Set<T>.minus(elements: Sequence<T>): Set<T> {\n    val result = LinkedHashSet<T>(this)\n    result.removeAll(elements)\n    return result\n}\n\n/**\n * Returns a set containing all elements of the original set  
except the given [element].\n */\n * The returned set preserves the element iteration order of the original set.\n\n * \n\n@kotlin.internal.InlineOnly\npublic inline fun <T> Set<T>.minusElement(element: T): Set<T> {\n    return  
minus(element)\n}\n\n/**\n * Returns a set containing all elements of the original set and then the given [element] if  
it isn't already in this set.\n */\n * The returned set preserves the element iteration order of the original set.\n\n * \n\npublic operator fun <T> Set<T>.plus(element: T): Set<T> {\n    val result =  
LinkedHashSet<T>(mapCapacity(size + 1))\n    result.addAll(this)\n    result.add(element)\n    return  
result\n}\n\n/**\n * Returns a set containing all elements of the original set and the given [elements] array,\n * which aren't already in this set.\n */\n * The returned set preserves the element iteration order of the original set.\n\n * \n\npublic operator fun <T> Set<T>.plus(elements: Array<out T>): Set<T> {\n    val result =  
LinkedHashSet<T>(mapCapacity(this.size + elements.size))\n    result.addAll(this)\n    result.addAll(elements)\n    return result\n}\n\n/**\n * Returns a set containing all elements of the original set and the given [elements]  
collection,\n * which aren't already in this set.\n */\n * The returned set preserves the element iteration order of the  
original set.\n * \n\npublic operator fun <T> Set<T>.plus(elements: Iterable<T>): Set<T> {\n    val result =  
LinkedHashSet<T>(mapCapacity(elements.collectionSizeOrNull()?.let { this.size + it } ?: this.size * 2))\n    result.addAll(this)\n    result.addAll(elements)\n    return result\n}\n\n/**\n * Returns a set containing all elements of  
the original set and the given [elements] sequence,\n * which aren't already in this set.\n */\n * The returned set
```

```

preserves the element iteration order of the original set.\n * \npublic operator fun <T> Set<T>.plus(elements:
Sequence<T>): Set<T> {\n    val result = LinkedHashMap<T>(mapCapacity(this.size * 2))\n    result.addAll(this)\n
result.addAll(elements)\n    return result\n}\n\n/**\n * Returns a set containing all elements of the original set and
then the given [element] if it isn't already in this set.\n * \n * The returned set preserves the element iteration
order of the original set.\n * \n@kotlin.internal.InlineOnly\npublic inline fun <T> Set<T>.plusElement(element: T):
Set<T> {\n    return plus(element)\n}\n\n"/\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n
*\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("\\StringsKt")\n\npackage
kotlin.text\n\n/\n// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt\n// See:
https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\nimport kotlin.random.*\n\n/**\n * Returns a
character at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this char
sequence.\n * \n * @sample samples.collections.Collections.Elements.elementAt\n * \npublic expect fun
CharSequence.elementAt(index: Int): Char\n\n/**\n * Returns a character at the given [index] or the result of calling
the [defaultValue] function if the [index] is out of bounds of this char sequence.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrElse\n * \n@kotlin.internal.InlineOnly\npublic inline fun
CharSequence.elementAtOrElse(index: Int, defaultValue: (Int) -> Char): Char {\n    return if (index >= 0 && index
<= lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns a character at the given [index] or `null` if
the [index] is out of bounds of this char sequence.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n * \n@kotlin.internal.InlineOnly\npublic inline fun
CharSequence.elementAtOrNull(index: Int): Char? {\n    return this.getOrNull(index)\n}\n\n/**\n * Returns the first
character matching the given [predicate], or `null` if no such character was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n * \n@kotlin.internal.InlineOnly\npublic inline fun
CharSequence.find(predicate: (Char) -> Boolean): Char? {\n    return firstOrNull(predicate)\n}\n\n/**\n * Returns
the last character matching the given [predicate], or `null` if no such character was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n * \n@kotlin.internal.InlineOnly\npublic inline fun
CharSequence.find(predicate: (Char) -> Boolean): Char? {\n    return lastOrNull(predicate)\n}\n\n/**\n *
Returns first character.\n * @throws [NoSuchElementException] if the char sequence is empty.\n * \npublic fun
CharSequence.first(): Char {\n    if (isEmpty())\n        throw NoSuchElementException("\\Char sequence is
empty.")\n    return this[0]\n}\n\n/**\n * Returns the first character matching the given [predicate].\n * @throws
[NoSuchElementException] if no such character is found.\n * \npublic inline fun CharSequence.first(predicate:
(Char) -> Boolean): Char {\n    for (element in this) if (predicate(element)) return element\n    throw
NoSuchElementException("\\Char sequence contains no character matching the predicate.")\n}\n\n/**\n * Returns
the first non-null value produced by [transform] function being applied to characters of this char sequence in
iteration order,\n * or throws [NoSuchElementException] if no non-null value was produced.\n * \n * @sample
samples.collections.Collections.Transformations.firstNotNullOf\n
*\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun <R : Any>
CharSequence.firstNotNullOf(transform: (Char) -> R?): R {\n    return firstNotNullOfOrNull(transform) ?: throw
NoSuchElementException("\\No element of the char sequence was transformed to a non-null value.")\n}\n\n/**\n *
Returns the first non-null value produced by [transform] function being applied to characters of this char sequence in
iteration order,\n * or `null` if no non-null value was produced.\n * \n * @sample
samples.collections.Collections.Transformations.firstNotNullOf\n
*\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun <R : Any>
CharSequence.firstNotNullOfOrNull(transform: (Char) -> R?): R? {\n    for (element in this) {\n        val result =
transform(element)\n        if (result != null) {\n            return result\n        }\n    }\n    return null\n}\n\n/**\n *
Returns the first character, or `null` if the char sequence is empty.\n * \npublic fun CharSequence.firstOrNull():
Char? {\n    return if (isEmpty()) null else this[0]\n}\n\n/**\n * Returns the first character matching the given
[predicate], or `null` if character was not found.\n * \npublic inline fun CharSequence.firstOrNull(predicate: (Char) -

```

```

> Boolean): Char? {\n  for (element in this) if (predicate(element)) return element\n  return null\n}\n\n/**\n *
Returns a character at the given [index] or the result of calling the [defaultValue] function if the [index] is out of
bounds of this char sequence.\n */\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.getOrElse(index:
Int, defaultValue: (Int) -> Char): Char {\n  return if (index >= 0 && index <= lastIndex) get(index) else
defaultValue(index)\n}\n\n/**\n * Returns a character at the given [index] or `null` if the [index] is out of bounds of
this char sequence.\n */\n * @sample samples.collections.Collections.Elements.getOrElse\n */\npublic fun
CharSequence.getOrElse(index: Int): Char? {\n  return if (index >= 0 && index <= lastIndex) get(index) else
null\n}\n\n/**\n * Returns index of the first character matching the given [predicate], or -1 if the char sequence does
not contain such character.\n */\npublic inline fun CharSequence.indexOfFirst(predicate: (Char) -> Boolean): Int {\n
for (index in indices) {\n  if (predicate(this[index])) {\n    return index\n  }\n }\n return -
1\n}\n\n/**\n * Returns index of the last character matching the given [predicate], or -1 if the char sequence does
not contain such character.\n */\npublic inline fun CharSequence.indexOfLast(predicate: (Char) -> Boolean): Int {\n
for (index in indices.reversed()) {\n  if (predicate(this[index])) {\n    return index\n  }\n }\n return -
1\n}\n\n/**\n * Returns the last character.\n */\n * @throws NoSuchElementException if the char sequence is
empty.\n */\n * @sample samples.text.Strings.last\n */\npublic fun CharSequence.last(): Char {\n  if (isEmpty())\n    throw NoSuchElementException("Char sequence is empty.")\n  return this[lastIndex]\n}\n\n/**\n * Returns the
last character matching the given [predicate].\n */\n * @throws NoSuchElementException if no such character is
found.\n */\n * @sample samples.text.Strings.last\n */\npublic inline fun CharSequence.last(predicate: (Char) ->
Boolean): Char {\n  for (index in this.indices.reversed()) {\n    val element = this[index]\n    if
(predicate(element)) return element\n  }\n  throw NoSuchElementException("Char sequence contains no
character matching the predicate.")\n}\n\n/**\n * Returns the last character, or `null` if the char sequence is
empty.\n */\n * @sample samples.text.Strings.last\n */\npublic fun CharSequence.lastOrNull(): Char? {\n  return if
(isEmpty()) null else this[length - 1]\n}\n\n/**\n * Returns the last character matching the given [predicate], or `null`
if no such character was found.\n */\n * @sample samples.text.Strings.last\n */\npublic inline fun
CharSequence.lastOrNull(predicate: (Char) -> Boolean): Char? {\n  for (index in this.indices.reversed()) {\n
val element = this[index]\n    if (predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns a
random character from this char sequence.\n */\n * @throws NoSuchElementException if this char sequence is
empty.\n */\n */\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.random(): Char
{\n  return random(Random)\n}\n\n/**\n * Returns a random character from this char sequence using the specified
source of randomness.\n */\n * @throws NoSuchElementException if this char sequence is empty.\n */\n */\n@SinceKotlin("1.3")\npublic fun CharSequence.random(random: Random): Char {\n  if (isEmpty())\n    throw NoSuchElementException("Char sequence is empty.")\n  return get(random.nextInt(length))\n}\n\n/**\n * Returns a random character from this char sequence, or `null` if this char sequence is empty.\n */\n */\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun CharSequence.randomOrNull(): Char? {\n  return randomOrNull(Random)\n}\n\n/**\n * Returns a
random character from this char sequence using the specified source of randomness, or `null` if this char sequence is
empty.\n */\n */\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
CharSequence.randomOrNull(random: Random): Char? {\n  if (isEmpty())\n    return null\n  return
get(random.nextInt(length))\n}\n\n/**\n * Returns the single character, or throws an exception if the char sequence
is empty or has more than one character.\n */\n */\npublic fun CharSequence.single(): Char {\n  return when (length)
{\n    0 -> throw NoSuchElementException("Char sequence is empty.")\n    1 -> this[0]\n    else -> throw
IllegalArgumentException("Char sequence has more than one element.")\n  }\n}\n\n/**\n * Returns the single
character matching the given [predicate], or throws exception if there is no or more than one matching character.\n */\n */\npublic inline fun CharSequence.single(predicate: (Char) -> Boolean): Char {\n  var single: Char? = null\n  var
found = false\n  for (element in this) {\n    if (predicate(element)) {\n      if (found) throw
IllegalArgumentException("Char sequence contains more than one matching element.")\n      single =
element\n      found = true\n    }\n  }\n  if (!found) throw NoSuchElementException("Char sequence
contains no character matching the predicate.")\n  @Suppress("UNCHECKED_CAST")\n  return single as

```

```

Char\n}\n\n/**\n * Returns single character, or `null` if the char sequence is empty or has more than one character.\n
*\n\npublic fun CharSequence.singleOrNull(): Char? {\n    return if (length == 1) this[0] else null\n}\n\n/**\n *
Returns the single character matching the given [predicate], or `null` if character was not found or more than one
character was found.\n
*\n\npublic inline fun CharSequence.singleOrNull(predicate: (Char) -> Boolean): Char? {\n
var single: Char? = null\n    var found = false\n    for (element in this) {\n        if (predicate(element)) {\n            if
(found) return null\n                single = element\n                found = true\n            }\n        }\n    } if (!found) return null\n    return
single\n}\n\n/**\n * Returns a subsequence of this char sequence with the first [n] characters removed.\n
*\n
*\n * @throws IllegalArgumentException if [n] is negative.\n
*\n
*\n * @sample samples.text.Strings.drop\n
*\n\npublic fun
CharSequence.drop(n: Int): CharSequence {\n    require(n >= 0) { \"Requested character count $n is less than zero.\"
}\n    return subSequence(n.coerceAtMost(length), length)\n}\n\n/**\n * Returns a string with the first [n] characters
removed.\n
*\n
*\n * @throws IllegalArgumentException if [n] is negative.\n
*\n
*\n * @sample
samples.text.Strings.drop\n
*\n\npublic fun String.drop(n: Int): String {\n    require(n >= 0) { \"Requested character
count $n is less than zero.\" }\n    return substring(n.coerceAtMost(length))\n}\n\n/**\n * Returns a subsequence of
this char sequence with the last [n] characters removed.\n
*\n
*\n * @throws IllegalArgumentException if [n] is
negative.\n
*\n
*\n * @sample samples.text.Strings.drop\n
*\n\npublic fun CharSequence.dropLast(n: Int):
CharSequence {\n    require(n >= 0) { \"Requested character count $n is less than zero.\" }\n    return take((length -
n).coerceAtLeast(0))\n}\n\n/**\n * Returns a string with the last [n] characters removed.\n
*\n
*\n * @throws
IllegalArgumentException if [n] is negative.\n
*\n
*\n * @sample samples.text.Strings.drop\n
*\n\npublic fun
String.dropLast(n: Int): String {\n    require(n >= 0) { \"Requested character count $n is less than zero.\" }\n    return
take((length - n).coerceAtLeast(0))\n}\n\n/**\n * Returns a subsequence of this char sequence containing all
characters except last characters that satisfy the given [predicate].\n
*\n
*\n * @sample samples.text.Strings.drop\n
*\n\npublic inline fun CharSequence.dropLastWhile(predicate: (Char) -> Boolean): CharSequence {\n    for (index in
lastIndex downTo 0)\n        if (!predicate(this[index]))\n            return subSequence(0, index + 1)\n    return
\"\"\n}\n\n/**\n * Returns a string containing all characters except last characters that satisfy the given [predicate].\n
*\n
*\n * @sample samples.text.Strings.drop\n
*\n\npublic inline fun String.dropLastWhile(predicate: (Char) ->
Boolean): String {\n    for (index in lastIndex downTo 0)\n        if (!predicate(this[index]))\n            return
substring(0, index + 1)\n    return \"\"\n}\n\n/**\n * Returns a subsequence of this char sequence containing all
characters except first characters that satisfy the given [predicate].\n
*\n
*\n * @sample samples.text.Strings.drop\n
*\n\npublic inline fun CharSequence.dropWhile(predicate: (Char) -> Boolean): CharSequence {\n    for (index in
this.indices)\n        if (!predicate(this[index]))\n            return subSequence(index, length)\n    return \"\"\n}\n\n/**\n * Returns a string containing all characters except first characters that satisfy the given [predicate].\n
*\n
*\n * @sample samples.text.Strings.drop\n
*\n\npublic inline fun String.dropWhile(predicate: (Char) -> Boolean): String
{\n    for (index in this.indices)\n        if (!predicate(this[index]))\n            return substring(index)\n    return
\"\"\n}\n\n/**\n * Returns a char sequence containing only those characters from the original char sequence that
match the given [predicate].\n
*\n
*\n * @sample samples.text.Strings.filter\n
*\n\npublic inline fun
CharSequence.filter(predicate: (Char) -> Boolean): CharSequence {\n    return filterTo(StringBuilder(),
predicate)\n}\n\n/**\n * Returns a string containing only those characters from the original string that match the
given [predicate].\n
*\n
*\n * @sample samples.text.Strings.filter\n
*\n\npublic inline fun String.filter(predicate: (Char) -
> Boolean): String {\n    return filterTo(StringBuilder(), predicate).toString()\n}\n\n/**\n * Returns a char sequence
containing only those characters from the original char sequence that match the given [predicate].\n
*\n
*\n * @param
[predicate] function that takes the index of a character and the character itself\n
*\n * and returns the result of predicate
evaluation on the character.\n
*\n
*\n * @sample samples.collections.Collections.Filtering.filterIndexed\n
*\n\npublic
inline fun CharSequence.filterIndexed(predicate: (index: Int, Char) -> Boolean): CharSequence {\n    return
filterIndexedTo(StringBuilder(), predicate)\n}\n\n/**\n * Returns a string containing only those characters from the
original string that match the given [predicate].\n
*\n
*\n * @param [predicate] function that takes the index of a character
and the character itself\n
*\n * and returns the result of predicate evaluation on the character.\n
*\n
*\n * @sample
samples.collections.Collections.Filtering.filterIndexed\n
*\n\npublic inline fun String.filterIndexed(predicate: (index:
Int, Char) -> Boolean): String {\n    return filterIndexedTo(StringBuilder(), predicate).toString()\n}\n\n/**\n
*

```

```

Appends all characters matching the given [predicate] to the given [destination].\n * @param [predicate] function
that takes the index of a character and the character itself\n * and returns the result of predicate evaluation on the
character.\n * \n * @sample samples.collections.Collections.Filtering.filterIndexedTo\n * \n\npublic inline fun <C :
Appendable> CharSequence.filterIndexedTo(destination: C, predicate: (index: Int, Char) -> Boolean): C {\n
forEachIndexed { index, element ->\n    if (predicate(index, element)) destination.append(element)\n    }\n
return destination\n}\n\n/**\n * Returns a char sequence containing only those characters from the original char
sequence that do not match the given [predicate].\n * \n * @sample samples.text.Strings.filterNot\n * \n\npublic inline
fun CharSequence.filterNot(predicate: (Char) -> Boolean): CharSequence {\n    return filterNotTo(StringBuilder(),
predicate)\n}\n\n/**\n * Returns a string containing only those characters from the original string that do not match
the given [predicate].\n * \n * @sample samples.text.Strings.filterNot\n * \n\npublic inline fun
String.filterNot(predicate: (Char) -> Boolean): String {\n    return filterNotTo(StringBuilder(),
predicate).toString()\n}\n\n/**\n * Appends all characters not matching the given [predicate] to the given
[destination].\n * \n * @sample samples.collections.Collections.Filtering.filterTo\n * \n\npublic inline fun <C :
Appendable> CharSequence.filterNotTo(destination: C, predicate: (Char) -> Boolean): C {\n    for (element in this)
if (!predicate(element)) destination.append(element)\n    return destination\n}\n\n/**\n * Appends all characters
matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n * \n\npublic inline fun <C : Appendable>
CharSequence.filterTo(destination: C, predicate: (Char) -> Boolean): C {\n    for (index in 0 until length) {\n        val
element = get(index)\n        if (predicate(element)) destination.append(element)\n    }\n    return
destination\n}\n\n/**\n * Returns a char sequence containing characters of the original char sequence at the
specified range of [indices].\n * \n\npublic fun CharSequence.slice(indices: IntRange): CharSequence {\n    if
(indices.isEmpty()) return ""\n    return subSequence(indices)\n}\n\n/**\n * Returns a string containing characters
of the original string at the specified range of [indices].\n * \n\npublic fun String.slice(indices: IntRange): String {\n
if (indices.isEmpty()) return ""\n    return substring(indices)\n}\n\n/**\n * Returns a char sequence containing
characters of the original char sequence at specified [indices].\n * \n\npublic fun CharSequence.slice(indices:
Iterable<Int>): CharSequence {\n    val size = indices.collectionSizeOrDefault(10)\n    if (size == 0) return ""\n
val result = StringBuilder(size)\n    for (i in indices) {\n        result.append(get(i))\n    }\n    return result\n}\n\n/**\n * Returns a string containing characters of the original string at specified [indices].\n
*\n * \n\n@kotlin.internal.InlineOnly\npublic inline fun String.slice(indices: Iterable<Int>): String {\n    return (this as
CharSequence).slice(indices).toString()\n}\n\n/**\n * Returns a subsequence of this char sequence containing the
first [n] characters from this char sequence, or the entire char sequence if this char sequence is shorter.\n * \n *
*\n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample samples.text.Strings.take\n * \n\npublic fun
CharSequence.take(n: Int): CharSequence {\n    require(n >= 0) { "Requested character count $n is less than zero."
}\n    return subSequence(0, n.coerceAtMost(length))\n}\n\n/**\n * Returns a string containing the first [n]
characters from this string, or the entire string if this string is shorter.\n * \n * \n * @throws IllegalArgumentException if
[n] is negative.\n * \n * @sample samples.text.Strings.take\n * \n\npublic fun String.take(n: Int): String {\n    require(
n >= 0) { "Requested character count $n is less than zero." }\n    return substring(0,
n.coerceAtMost(length))\n}\n\n/**\n * Returns a subsequence of this char sequence containing the last [n]
characters from this char sequence, or the entire char sequence if this char sequence is shorter.\n * \n *
*\n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample samples.text.Strings.take\n * \n\npublic fun
CharSequence.takeLast(n: Int): CharSequence {\n    require(n >= 0) { "Requested character count $n is less than
zero." }\n    val length = length\n    return subSequence(length - n.coerceAtMost(length), length)\n}\n\n/**\n *
Returns a string containing the last [n] characters from this string, or the entire string if this string is shorter.\n *
*\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample samples.text.Strings.take\n * \n\npublic fun
String.takeLast(n: Int): String {\n    require(n >= 0) { "Requested character count $n is less than zero." }\n    val
length = length\n    return substring(length - n.coerceAtMost(length))\n}\n\n/**\n * Returns a subsequence of this
char sequence containing last characters that satisfy the given [predicate].\n * \n * @sample
samples.text.Strings.take\n * \n\npublic inline fun CharSequence.takeLastWhile(predicate: (Char) -> Boolean):

```

```

CharSequence {\n  for (index in lastIndex downTo 0) {\n    if (!predicate(this[index])) {\n      return
subSequence(index + 1, length)\n    }\n  }\n  return subSequence(0, length)\n}\n\n/**\n * Returns a string
containing last characters that satisfy the given [predicate].\n * \n * @sample samples.text.Strings.take\n */\npublic
inline fun String.takeLastWhile(predicate: (Char) -> Boolean): String {\n  for (index in lastIndex downTo 0) {\n
if (!predicate(this[index])) {\n    return substring(index + 1)\n  }\n  return this\n}\n\n/**\n * Returns
a subsequence of this char sequence containing the first characters that satisfy the given [predicate].\n * \n *
@sample samples.text.Strings.take\n */\npublic inline fun CharSequence.takeWhile(predicate: (Char) -> Boolean):
CharSequence {\n  for (index in 0 until length)\n    if (!predicate(get(index))) {\n      return subSequence(0,
index)\n    }\n  return subSequence(0, length)\n}\n\n/**\n * Returns a string containing the first characters that
satisfy the given [predicate].\n * \n * @sample samples.text.Strings.take\n */\npublic inline fun
String.takeWhile(predicate: (Char) -> Boolean): String {\n  for (index in 0 until length)\n    if
(!predicate(get(index))) {\n      return substring(0, index)\n    }\n  return this\n}\n\n/**\n * Returns a char
sequence with characters in reversed order.\n */\npublic fun CharSequence.reversed(): CharSequence {\n  return
StringBuilder(this).reverse()\n}\n\n/**\n * Returns a string with characters in reversed order.\n
*/\n@kotlin.internal.InlineOnly\npublic inline fun String.reversed(): String {\n  return (this as
CharSequence).reversed().toString()\n}\n\n/**\n * Returns a [Map] containing key-value pairs provided by
[transform] function\n * applied to characters of the given char sequence.\n * \n * If any of two pairs would have the
same key the last one gets added to the map.\n * \n * The returned map preserves the entry iteration order of the
original char sequence.\n * \n * @sample samples.text.Strings.associate\n */\npublic inline fun <K, V>
CharSequence.associate(transform: (Char) -> Pair<K, V>): Map<K, V> {\n  val capacity =
mapCapacity(length).coerceAtLeast(16)\n  return associateTo(LinkedHashMap<K, V>(capacity),
transform)\n}\n\n/**\n * Returns a [Map] containing the characters from the given char sequence indexed by the
key\n * returned from [keySelector] function applied to each character.\n * \n * If any two characters would have the
same key returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves the entry
iteration order of the original char sequence.\n * \n * @sample samples.text.Strings.associateBy\n */\npublic inline
fun <K> CharSequence.associateBy(keySelector: (Char) -> K): Map<K, Char> {\n  val capacity =
mapCapacity(length).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, Char>(capacity),
keySelector)\n}\n\n/**\n * Returns a [Map] containing the values provided by [valueTransform] and indexed by
[keySelector] functions applied to characters of the given char sequence.\n * \n * If any two characters would have
the same key returned by [keySelector] the last one gets added to the map.\n * \n * The returned map preserves the
entry iteration order of the original char sequence.\n * \n * @sample
samples.text.Strings.associateByWithValueTransform\n */\npublic inline fun <K, V>
CharSequence.associateBy(keySelector: (Char) -> K, valueTransform: (Char) -> V): Map<K, V> {\n  val capacity
= mapCapacity(length).coerceAtLeast(16)\n  return associateByTo(LinkedHashMap<K, V>(capacity),
keySelector, valueTransform)\n}\n\n/**\n * Populates and returns the [destination] mutable map with key-value
pairs,\n * where key is provided by the [keySelector] function applied to each character of the given char sequence\n
* and value is the character itself.\n * \n * If any two characters would have the same key returned by [keySelector]
the last one gets added to the map.\n * \n * @sample samples.text.Strings.associateByTo\n */\npublic inline fun <K,
M : MutableMap<in K, in Char>> CharSequence.associateByTo(destination: M, keySelector: (Char) -> K): M {\n
for (element in this) {\n  destination.put(keySelector(element), element)\n }\n  return destination\n}\n\n/**\n
* Populates and returns the [destination] mutable map with key-value pairs,\n * where key is provided by the
[keySelector] function and\n * and value is provided by the [valueTransform] function applied to characters of the
given char sequence.\n * \n * If any two characters would have the same key returned by [keySelector] the last one
gets added to the map.\n * \n * @sample samples.text.Strings.associateByToWithValueTransform\n */\npublic
inline fun <K, V, M : MutableMap<in K, in V>> CharSequence.associateByTo(destination: M, keySelector: (Char)
-> K, valueTransform: (Char) -> V): M {\n  for (element in this) {\n    destination.put(keySelector(element),
valueTransform(element))\n  }\n  return destination\n}\n\n/**\n * Populates and returns the [destination] mutable
map with key-value pairs\n * provided by [transform] function applied to each character of the given char

```

```

sequence.\n * \n * If any of two pairs would have the same key the last one gets added to the map.\n * \n * @sample
samples.text.Strings.associateTo\n * \n\npublic inline fun <K, V, M : MutableMap<in K, in V>>
CharSequence.associateTo(destination: M, transform: (Char) -> Pair<K, V>): M {\n  for (element in this) {\n
destination += transform(element)\n  }\n  return destination\n}\n\n/**\n * Returns a [Map] where keys are
characters from the given char sequence and values are\n * produced by the [valueSelector] function applied to each
character.\n * \n * If any two characters are equal, the last one gets added to the map.\n * \n * The returned map
preserves the entry iteration order of the original char sequence.\n * \n * @sample
samples.text.Strings.associateWith\n * \n\n@SinceKotlin("1.3")\n\npublic inline fun <V>
CharSequence.associateWith(valueSelector: (Char) -> V): Map<Char, V> {\n  val result = LinkedHashMap<Char,
V>(mapCapacity(length.coerceAtMost(128)).coerceAtLeast(16))\n  return associateWithTo(result,
valueSelector)\n}\n\n/**\n * Populates and returns the [destination] mutable map with key-value pairs for each
character of the given char sequence,\n * where key is the character itself and value is provided by the
[valueSelector] function applied to that key.\n * \n * If any two characters are equal, the last one overwrites the
former value in the map.\n * \n * @sample samples.text.Strings.associateWithTo\n
*\n\n@SinceKotlin("1.3")\n\npublic inline fun <V, M : MutableMap<in Char, in V>>
CharSequence.associateWithTo(destination: M, valueSelector: (Char) -> V): M {\n  for (element in this) {\n
destination.put(element, valueSelector(element))\n  }\n  return destination\n}\n\n/**\n * Appends all characters to
the given [destination] collection.\n * \n\npublic fun <C : MutableCollection<in Char>>
CharSequence.toCollection(destination: C): C {\n  for (item in this) {\n  destination.add(item)\n  }\n  return
destination\n}\n\n/**\n * Returns a new [HashSet] of all characters.\n * \n\npublic fun CharSequence.toHashSet():
HashSet<Char> {\n  return toCollection(HashSet<Char>(mapCapacity(length.coerceAtMost(128))))\n}\n\n/**\n *
Returns a [List] containing all characters.\n * \n\npublic fun CharSequence.toList(): List<Char> {\n  return when
(length) {\n  0 -> emptyList()\n  1 -> listOf(this[0])\n  else -> this.toMutableList()\n  }\n}\n\n/**\n *
Returns a new [MutableList] filled with all characters of this char sequence.\n * \n\npublic fun
CharSequence.toMutableList(): MutableList<Char> {\n  return toCollection(ArrayList<Char>(length))\n}\n\n/**\n *
Returns a [Set] of all characters.\n * \n * The returned set preserves the element iteration order of the original char
sequence.\n * \n\npublic fun CharSequence.toSet(): Set<Char> {\n  return when (length) {\n  0 -> emptySet()\n
1 -> setOf(this[0])\n  else -> toCollection(LinkedHashSet<Char>(mapCapacity(length.coerceAtMost(128))))\n
}\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on
each character of original char sequence.\n * \n * @sample
samples.collections.Collections.Transformations.flatMap\n * \n\npublic inline fun <R>
CharSequence.flatMap(transform: (Char) -> Iterable<R>): List<R> {\n  return flatMapTo(ArrayList<R>(),
transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being
invoked on each character\n * and its index in the original char sequence.\n * \n * @sample
samples.collections.Collections.Transformations.flatMapIndexed\n
*\n\n@SinceKotlin("1.4")\n\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n\n@OverloadResolution
ByLambdaReturnType\n\n@kotlin.jvm.JvmName("flatMapIndexedIterable")\n\n@kotlin.internal.InlineOnly\n\npublic
inline fun <R> CharSequence.flatMapIndexed(transform: (index: Int, Char) -> Iterable<R>): List<R> {\n  return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Appends all elements yielded from results of
[transform] function being invoked on each character\n * and its index in the original char sequence, to the given
[destination].\n * \n\n*\n\n@SinceKotlin("1.4")\n\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n\n@OverloadResolution
ByLambdaReturnType\n\n@kotlin.jvm.JvmName("flatMapIndexedIterableTo")\n\n@kotlin.internal.InlineOnly\n\npubli
c inline fun <R, C : MutableCollection<in R>> CharSequence.flatMapIndexedTo(destination: C, transform: (index:
Int, Char) -> Iterable<R>): C {\n  var index = 0\n  for (element in this) {\n  val list = transform(index++,
element)\n  destination.addAll(list)\n  }\n  return destination\n}\n\n/**\n * Appends all elements yielded from
results of [transform] function being invoked on each character of original char sequence, to the given
[destination].\n * \n\npublic inline fun <R, C : MutableCollection<in R>> CharSequence.flatMapTo(destination: C,

```

```

transform: (Char) -> Iterable<R>): C {
    for (element in this) {
        val list = transform(element)
        destination.addAll(list)
    }
    return destination
}

```

Groups characters of the original char sequence by the key returned by the given [keySelector] function applied to each character and returns a map where each group key is associated with a list of corresponding characters. The returned map preserves the entry iteration order of the keys produced from the original char sequence.

@sample

```

samples.collections.Collections.Transformations.groupBy

```

```

CharSequence.groupBy(keySelector: (Char) -> K): Map<K, List<Char>> {
    return
    groupByTo(LinkedHashMap<K, MutableList<Char>>(), keySelector)
}

```

Groups values returned by the [valueTransform] function applied to each character of the original char sequence by the key returned by the given [keySelector] function applied to the character and returns a map where each group key is associated with a list of corresponding values. The returned map preserves the entry iteration order of the keys produced from the original char sequence.

@sample

```

samples.collections.Collections.Transformations.groupByKeysAndValues

```

```

CharSequence.groupBy(keySelector: (Char) -> K, valueTransform: (Char) -> V): Map<K, List<V>> {
    return
    groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)
}

```

Groups characters of the original char sequence by the key returned by the given [keySelector] function applied to each character and puts to the [destination] map each group key associated with a list of corresponding characters.

@return The [destination] map.

@sample

```

samples.collections.Collections.Transformations.groupBy

```

```

CharSequence.groupByTo(destination: M, keySelector: (Char) -> K): M {
    for (element in this) {
        val key = keySelector(element)
        val list =
        destination.getOrPut(key) { ArrayList<Char>() }
        list.add(element)
    }
    return destination
}

```

Groups values returned by the [valueTransform] function applied to each character of the original char sequence by the key returned by the given [keySelector] function applied to the character and puts to the [destination] map each group key associated with a list of corresponding values.

@return The [destination] map.

@sample

```

samples.collections.Collections.Transformations.groupByKeysAndValues

```

```

CharSequence.groupByTo(destination: M, keySelector: (Char) -> K,
valueTransform: (Char) -> V): M {
    for (element in this) {
        val key = keySelector(element)
        val list =
        destination.getOrPut(key) { ArrayList<V>() }
        list.add(valueTransform(element))
    }
    return
    destination
}

```

Creates a [Grouping] source from a char sequence to be used later with one of group-and-fold operations using the specified [keySelector] function to extract a key from each character.

@sample

```

samples.collections.Grouping.groupingByEachCount

```

```

CharSequence.groupingBy(crossinline keySelector: (Char) -> K): Grouping<Char, K> {
    return object :
    Grouping<Char, K> {
        override fun sourceIterator(): Iterator<Char> = this@groupingBy.iterator()
        override fun keyOf(element: Char): K = keySelector(element)
    }
}

```

Returns a list containing the results of applying the given [transform] function to each character in the original char sequence.

@sample

```

samples.text.Strings.map

```

```

CharSequence.map(transform: (Char) -> R):
List<R> {
    return mapTo(ArrayList<R>(length), transform)
}

```

Returns a list containing the results of applying the given [transform] function to each character and its index in the original char sequence.

@param [transform] function that takes the index of a character and the character itself and returns the result of the transform applied to the character.

```

CharSequence.mapIndexed(transform: (index:
Int, Char) -> R): List<R> {
    return mapIndexedTo(ArrayList<R>(length), transform)
}

```

Returns a list containing only the non-null results of applying the given [transform] function to each character and its index in the original char sequence.

@param [transform] function that takes the index of a character and the character itself and returns the result of the transform applied to the character.

```

CharSequence.mapIndexedNotNull(transform: (index: Int, Char) -> R?): List<R> {
    return
    mapIndexedNotNullTo(ArrayList<R>(), transform)
}

```

Applies the given [transform] function to each character and its index in the original char sequence and appends only the non-null results to the given [destination].

@param [transform] function that takes the index of a character and the character itself and

returns the result of the transform applied to the character.

```

public inline fun <R : Any, C :
MutableCollection<in R>> CharSequence.mapIndexedNotNullTo(destination: C, transform: (index: Int, Char) ->
R?): C {
    for<in> indexed { index, element -> transform(index, element)?.let { destination.add(it) } }
    return destination
}

```

\* Applies the given [transform] function to each character and its index in the original char sequence and appends the results to the given [destination].

\* @param [transform] function that takes the index of a character and the character itself and returns the result of the transform applied to the character.

```

public inline fun <R, C : MutableCollection<in R>> CharSequence.mapIndexedTo(destination: C, transform:
(index: Int, Char) -> R): C {
    var index = 0
    for<in> (item in this) destination.add(transform(index++, item))
    return destination
}

```

\* Returns a list containing only the non-null results of applying the given [transform] function to each character in the original char sequence.

\* @sample samples.collections.Collections.Transformations.mapNotNull

```

public inline fun <R : Any> CharSequence.mapNotNull(transform: (Char) -> R?): List<R> {
    return mapNotNullTo(ArrayList<R>(), transform)
}

```

\* Applies the given [transform] function to each character in the original char sequence and appends only the non-null results to the given [destination].

```

public inline fun <R : Any, C : MutableCollection<in R>> CharSequence.mapNotNullTo(destination: C, transform: (Char) -> R?): C {
    for<in> element { element -> transform(element)?.let { destination.add(it) } }
    return destination
}

```

\* Applies the given [transform] function to each character of the original char sequence and appends the results to the given [destination].

```

public inline fun <R, C : MutableCollection<in R>> CharSequence.mapTo(destination: C, transform: (Char) -> R): C {
    for<in> (item in this) destination.add(transform(item))
    return destination
}

```

\* Returns a lazy [Iterable] that wraps each character of the original char sequence into an [IndexedValue] containing the index of that character and the character itself.

```

public fun CharSequence.withIndex(): Iterable<IndexedValue<Char>> {
    return IndexingIterable { iterator() }
}

```

\* Returns `true` if all characters match the given [predicate].

\* @sample samples.collections.Collections.Aggregates.all

```

public inline fun CharSequence.all(predicate: (Char) -> Boolean): Boolean {
    for<in> (element in this) if (!predicate(element)) return false
    return true
}

```

\* Returns `true` if char sequence has at least one character.

\* @sample samples.collections.Collections.Aggregates.any

```

public fun CharSequence.any(): Boolean {
    return !isEmpty()
}

```

\* Returns `true` if at least one character matches the given [predicate].

\* @sample samples.collections.Collections.Aggregates.anyWithPredicate

```

public inline fun CharSequence.any(predicate: (Char) -> Boolean): Boolean {
    for<in> (element in this) if (predicate(element)) return true
    return false
}

```

\* Returns the length of this char sequence.

```

@kotlin.internal.InlineOnly
public inline fun CharSequence.count(): Int {
    return length
}

```

\* Returns the number of characters matching the given [predicate].

```

public inline fun CharSequence.count(predicate: (Char) -> Boolean): Int {
    var count = 0
    for<in> (element in this) if (predicate(element)) ++count
    return count
}

```

\* Accumulates value starting with [initial] value and applying [operation] from left to right to current accumulator value and each character.

\* @sample samples.collections.Collections.Aggregates.fold

\* Returns the specified [initial] value if the char sequence is empty.

\* @param [operation] function that takes current accumulator value and a character, and calculates the next accumulator value.

```

public inline fun <R> CharSequence.fold(initial: R, operation: (acc: R, Char) -> R): R {
    var accumulator = initial
    for<in> (element in this) accumulator = operation(accumulator, element)
    return accumulator
}

```

\* Accumulates value starting with [initial] value and applying [operation] from left to right to current accumulator value and each character with its index in the original char sequence.

\* @sample samples.collections.Collections.Aggregates.foldIndexed

\* Returns the specified [initial] value if the char sequence is empty.

\* @param [operation] function that takes the index of a character, current accumulator value and the character itself, and calculates the next accumulator value.

```

public inline fun <R> CharSequence.foldIndexed(initial: R, operation: (index: Int, acc: R, Char) -> R): R {
    var index = 0
    var accumulator = initial
    for<in> (element in this) accumulator = operation(index++, accumulator, element)
    return accumulator
}

```

\* Accumulates value starting with [initial] value and applying [operation] from right to left to each character and current accumulator value.

\* @sample samples.collections.Collections.Aggregates.foldRight

\* Returns the specified [initial] value if the char sequence is empty.

\* @param [operation] function that takes a character and current accumulator value, and

```

calculates the next accumulator value.\n */\npublic inline fun <R> CharSequence.foldRight(initial: R, operation:
(Char, acc: R) -> R): R {\n    var index = lastIndex\n    var accumulator = initial\n    while (index >= 0) {\n
accumulator = operation(get(index--), accumulator)\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value
starting with [initial] value and applying [operation] from right to left\n * to each character with its index in the
original char sequence and current accumulator value.\n * \n * Returns the specified [initial] value if the char
sequence is empty.\n * \n * @param [operation] function that takes the index of a character, the character itself\n *
and current accumulator value, and calculates the next accumulator value.\n */\npublic inline fun <R>
CharSequence.foldRightIndexed(initial: R, operation: (index: Int, Char, acc: R) -> R): R {\n    var index =
lastIndex\n    var accumulator = initial\n    while (index >= 0) {\n        accumulator = operation(index, get(index),
accumulator)\n        --index\n    }\n    return accumulator\n}\n\n/**\n * Performs the given [action] on each
character.\n */\npublic inline fun CharSequence.forEach(action: (Char) -> Unit): Unit {\n    for (element in this)
action(element)\n}\n\n/**\n * Performs the given [action] on each character, providing sequential index with the
character.\n * @param [action] function that takes the index of a character and the character itself\n * and performs
the action on the character.\n */\npublic inline fun CharSequence.forEachIndexed(action: (index: Int, Char) -> Unit):
Unit {\n    var index = 0\n    for (item in this) action(index++, item)\n}\n\n@Deprecated("Use maxOrNull
instead.", ReplaceWith("this.maxOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic fun CharSequence.max(): Char? {\n    return
maxOrNull()\n}\n\n@Deprecated("Use maxByOrNull instead.",
ReplaceWith("this.maxByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>> CharSequence.maxBy(selector: (Char) ->
R): Char? {\n    return maxByOrNull(selector)\n}\n\n/**\n * Returns the first character yielding the largest value of
the given function or `null` if there are no characters.\n * \n * @sample
samples.collections.Collections.Aggregates.maxByOrNull\n */\n@SinceKotlin("1.4")\npublic inline fun <R :
Comparable<R>> CharSequence.maxByOrNull(selector: (Char) -> R): Char? {\n    if (isEmpty()) return null\n    var
maxElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return maxElem\n    var maxValue =
selector(maxElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (maxValue < v)
{\n            maxElem = e\n            maxValue = v\n        }\n    }\n    return maxElem\n}\n\n/**\n * Returns the largest
value among all values produced by [selector] function\n * applied to each character in the char sequence.\n * \n * If
any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws
NoSuchElementException if the char sequence is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.maxOf(selector: (Char) ->
Double): Double {\n    if (isEmpty()) throw NoSuchElementException()\n    var maxValue = selector(this[0])\n    for
(i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return
maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each character in the char sequence.\n * \n * If any of values produced by [selector] function is `NaN`, the returned
result is `NaN`.\n * \n * @throws NoSuchElementException if the char sequence is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.maxOf(selector: (Char) ->
Float): Float {\n    if (isEmpty()) throw NoSuchElementException()\n    var maxValue = selector(this[0])\n    for (i
in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return
maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each character in the char sequence.\n * \n * @throws NoSuchElementException if the char sequence is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
CharSequence.maxOf(selector: (Char) -> R): R {\n    if (isEmpty()) throw NoSuchElementException()\n    var
maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if (maxValue < v) {\n
maxValue = v\n        }\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values

```

produced by [selector] function\n \* applied to each character in the char sequence or `null` if there are no characters.\n \* \n \* If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.maxOfOrNull(selector:
(Char) -> Double): Double? {\n if (isEmpty()) return null\n var maxValue = selector(this[0])\n for (i in
1..lastIndex) {\n val v = selector(this[i])\n maxValue = maxOf(maxValue, v)\n }\n return
maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each character in the char sequence or `null` if there are no characters.\n * \n * If any of values produced by
[selector] function is `NaN`, the returned result is `NaN`.\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.maxOfOrNull(selector:
(Char) -> Float): Float? {\n if (isEmpty()) return null\n var maxValue = selector(this[0])\n for (i in
1..lastIndex) {\n val v = selector(this[i])\n maxValue = maxOf(maxValue, v)\n }\n return
maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to
each character in the char sequence or `null` if there are no characters.\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>>
CharSequence.maxOfOrNull(selector: (Char) -> R): R? {\n if (isEmpty()) return null\n var maxValue =
selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if (maxValue < v) {\n
maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n * Returns the largest value according to the provided
[comparator]\n * among all values produced by [selector] function applied to each character in the char sequence.\n
* \n * @throws NoSuchElementException if the char sequence is empty.\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R> CharSequence.maxOfWith(comparator:
Comparator<in R>, selector: (Char) -> R): R {\n if (isEmpty()) throw NoSuchElementException()\n var
maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if
(comparator.compare(maxValue, v) < 0) {\n maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n * Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each character in the char sequence or `null` if there are no characters.\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R>
CharSequence.maxOfWithOrNull(comparator: Comparator<in R>, selector: (Char) -> R): R? {\n if (isEmpty())
return null\n var maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if
(comparator.compare(maxValue, v) < 0) {\n maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n * Returns the largest character or `null` if there are no characters.\n

```

```

*\n@SinceKotlin("1.4")\npublic fun
CharSequence.maxOrNull(): Char? {\n if (isEmpty()) return null\n var max = this[0]\n for (i in 1..lastIndex)
{\n val e = this[i]\n if (max < e) max = e\n }\n return max\n}\n\n@Deprecated("Use maxWithOrNull
instead.", ReplaceWith("this.maxWithOrNull(comparator)"))\n@DeprecatedSinceKotlin(warningSince = "1.4",
errorSince = "1.5", hiddenSince = "1.6")\npublic fun CharSequence.maxWith(comparator: Comparator<in
Char>): Char? {\n return maxWithOrNull(comparator)\n}\n\n/**\n * Returns the first character having the largest
value according to the provided [comparator] or `null` if there are no characters.\n

```

```

*\n@SinceKotlin("1.4")\npublic fun CharSequence.maxWithOrNull(comparator: Comparator<in Char>): Char?
{\n if (isEmpty()) return null\n var max = this[0]\n for (i in 1..lastIndex) {\n val e = this[i]\n if
(comparator.compare(max, e) < 0) max = e\n }\n return max\n}\n\n@Deprecated("Use minOrNull instead.",
ReplaceWith("this.minOrNull()"))\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\npublic fun CharSequence.min(): Char? {\n return minOrNull()\n}\n\n@Deprecated("Use
minByOrNull instead.", ReplaceWith("this.minByOrNull(selector)"))\n@DeprecatedSinceKotlin(warningSince =
"1.4", errorSince = "1.5", hiddenSince = "1.6")\npublic inline fun <R : Comparable<R>>

```

```

CharSequence.minBy(selector: (Char) -> R): Char? {\n  return minByOrNull(selector)\n}\n\n/**\n * Returns the first character yielding the smallest value of the given function or `null` if there are no characters.\n * \n * @sample samples.collections.Collections.Aggregates.minByOrNull\n */\n@SinceKotlin("1.4")\npublic inline fun <R : Comparable<R>> CharSequence.minByOrNull(selector: (Char) -> R): Char? {\n  if (isEmpty()) return null\n  var minElem = this[0]\n  val lastIndex = this.lastIndex\n  if (lastIndex == 0) return minElem\n  var minValue = selector(minElem)\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    val v = selector(e)\n    if (minValue > v) {\n      minElem = e\n      minValue = v\n    }\n  }\n  return minElem\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each character in the char sequence.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the char sequence is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.minOf(selector: (Char) -> Double): Double {\n  if (isEmpty()) throw NoSuchElementException()\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each character in the char sequence.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the char sequence is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.minOf(selector: (Char) -> Float): Float {\n  if (isEmpty()) throw NoSuchElementException()\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each character in the char sequence.\n * \n * @throws NoSuchElementException if the char sequence is empty.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>> CharSequence.minOf(selector: (Char) -> R): R {\n  if (isEmpty()) throw NoSuchElementException()\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (minValue > v) {\n      minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each character in the char sequence or `null` if there are no characters.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.minOfOrNull(selector: (Char) -> Double): Double? {\n  if (isEmpty()) return null\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each character in the char sequence or `null` if there are no characters.\n * \n * If any of values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.minOfOrNull(selector: (Char) -> Float): Float? {\n  if (isEmpty()) return null\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each character in the char sequence or `null` if there are no characters.\n */\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@kotlin.internal.InlineOnly\npublic inline fun <R : Comparable<R>> CharSequence.minOfOrNull(selector: (Char) -> R): R? {\n  if (isEmpty()) return null\n  var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (minValue > v) {\n      minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value according to the provided

```

[comparator] among all values produced by [selector] function applied to each character in the char sequence.

\* @throws NoSuchElementException if the char sequence is empty.

```

* Since Kotlin("1.4")
* OptIn(kotlin.experimental.ExperimentalTypeInference::class)
* OverloadResolutionByLambdaReturnType
* kotlin.internal.InlineOnly
public inline fun <R> CharSequence.minOfWith(comparator:
Comparator<in R>, selector: (Char) -> R): R {
    if (isEmpty()) throw NoSuchElementException()
    var min = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        if (comparator.compare(min, v) > 0) {
            min = v
        }
    }
    return min
}

```

Returns the smallest value according to the provided [comparator] among all values produced by [selector] function applied to each character in the char sequence or `null` if there are no characters.

```

* Since Kotlin("1.4")
* OptIn(kotlin.experimental.ExperimentalTypeInference::class)
* OverloadResolutionByLambdaReturnType
* kotlin.internal.InlineOnly
public inline fun <R>
CharSequence.minOfOrNull(comparator: Comparator<in R>, selector: (Char) -> R): R? {
    if (isEmpty())
        return null
    var min = selector(this[0])
    for (i in 1..lastIndex) {
        val v = selector(this[i])
        if (comparator.compare(min, v) > 0) {
            min = v
        }
    }
    return min
}

```

Returns the smallest character or `null` if there are no characters.

```

* Since Kotlin("1.4")
public fun
CharSequence.minOrNull(): Char? {
    if (isEmpty()) return null
    var min = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (min > e) min = e
    }
    return min
}

```

**Deprecated** ("Use minWithOrNull instead.", ReplaceWith("this.minWithOrNull(comparator)"))

**Deprecated Since Kotlin** (warningSince = "1.4", errorSince = "1.5", hiddenSince = "1.6")

```

public fun CharSequence.minWith(comparator: Comparator<in
Char>): Char? {
    return minWithOrNull(comparator)
}

```

Returns the first character having the smallest value according to the provided [comparator] or `null` if there are no characters.

```

* Since Kotlin("1.4")
public fun CharSequence.minWithOrNull(comparator: Comparator<in Char>): Char?
{
    if (isEmpty()) return null
    var min = this[0]
    for (i in 1..lastIndex) {
        val e = this[i]
        if (comparator.compare(min, e) > 0) min = e
    }
    return min
}

```

Returns `true` if the char sequence has no characters.

\* @sample samples.collections.Collections.Aggregates.none

```

* public fun
CharSequence.none(): Boolean {
    return isEmpty()
}

```

Returns `true` if no characters match the given [predicate].

\* @sample samples.collections.Collections.Aggregates.noneWithPredicate

```

* public inline fun
CharSequence.none(predicate: (Char) -> Boolean): Boolean {
    for (element in this) if (predicate(element)) return
false
    return true
}

```

Performs the given [action] on each character and returns the char sequence itself afterwards.

```

* Since Kotlin("1.1")
public inline fun <S : CharSequence> S.onEach(action: (Char) -> Unit): S
{
    return apply { for (element in this) action(element) }
}

```

Performs the given [action] on each character, providing sequential index with the character, and returns the char sequence itself afterwards.

\* @param [action] function that takes the index of a character and the character itself and performs the action on the character.

```

* Since Kotlin("1.4")
public inline fun <S : CharSequence> S.onEachIndexed(action: (index:
Int, Char) -> Unit): S {
    return apply { forEachIndexed(action) }
}

```

Accumulates value starting with the first character and applying [operation] from left to right to current accumulator value and each character.

\* Throws an exception if this char sequence is empty. If the char sequence can be empty in an expected way, please use [reduceOrNull] instead. It returns `null` when its receiver is empty.

\* @param [operation] function that takes current accumulator value and a character, and calculates the next accumulator value.

```

* @sample samples.collections.Collections.Aggregates.reduce
* public inline fun
CharSequence.reduce(operation: (acc: Char, Char) -> Char): Char {
    if (isEmpty())
        throw
UnsupportedOperationException("Empty char sequence can't be reduced.")
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator = operation(accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first character and applying [operation] from left to right to current accumulator value and each character with its index in the original char sequence.

\* Throws an exception if this char sequence is empty. If the char sequence can be empty in an expected way, please use [reduceIndexedOrNull] instead. It returns `null` when its receiver is empty.

\* @param [operation] function that takes the index of a character, current accumulator value and the character itself, and calculates the

```

next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduce\n * \n\npublic inline fun
CharSequence.reduceIndexed(operation: (index: Int, acc: Char, Char) -> Char): Char {\n if (isEmpty())\n
throw UnsupportedOperationException("Empty char sequence can't be reduced.")\n var accumulator = this[0]\n
for (index in 1..lastIndex) {\n accumulator = operation(index, accumulator, this[index])\n }\n return
accumulator}\n\n\n/**\n * Accumulates value starting with the first character and applying [operation] from left to
right\n * to current accumulator value and each character with its index in the original char sequence.\n * \n *
Returns `null` if the char sequence is empty.\n * \n * @param [operation] function that takes the index of a
character, current accumulator value and the character itself,\n * and calculates the next accumulator value.\n * \n *
@sample samples.collections.Collections.Aggregates.reduceOrNull\n * \n\n@SinceKotlin("1.4")\npublic inline fun
CharSequence.reduceIndexedOrNull(operation: (index: Int, acc: Char, Char) -> Char): Char? {\n if (isEmpty())\n
return null\n var accumulator = this[0]\n for (index in 1..lastIndex) {\n accumulator = operation(index,
accumulator, this[index])\n }\n return accumulator}\n\n\n/**\n * Accumulates value starting with the first
character and applying [operation] from left to right\n * to current accumulator value and each character.\n * \n *
Returns `null` if the char sequence is empty.\n * \n * @param [operation] function that takes current accumulator
value and a character,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceOrNull\n
*\n\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
CharSequence.reduceOrNull(operation: (acc: Char, Char) -> Char): Char? {\n if (isEmpty())\n return null\n
var accumulator = this[0]\n for (index in 1..lastIndex) {\n accumulator = operation(accumulator, this[index])\n
}\n return accumulator}\n\n\n/**\n * Accumulates value starting with the last character and applying [operation]
from right to left\n * to each character and current accumulator value.\n * \n * Throws an exception if this char
sequence is empty. If the char sequence can be empty in an expected way,\n * please use [reduceRightOrNull]
instead. It returns `null` when its receiver is empty.\n * \n * @param [operation] function that takes a character and
current accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n * \n\npublic inline fun
CharSequence.reduceRight(operation: (Char, acc: Char) -> Char): Char {\n var index = lastIndex\n if (index < 0)
throw UnsupportedOperationException("Empty char sequence can't be reduced.")\n var accumulator = get(index-
-)\n while (index >= 0) {\n accumulator = operation(get(index--), accumulator)\n }\n return
accumulator}\n\n\n/**\n * Accumulates value starting with the last character and applying [operation] from right to
left\n * to each character with its index in the original char sequence and current accumulator value.\n * \n * Throws
an exception if this char sequence is empty. If the char sequence can be empty in an expected way,\n * please use
[reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n * @param [operation]
function that takes the index of a character, the character itself and current accumulator value,\n * and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceRight\n * \n\npublic
inline fun CharSequence.reduceRightIndexed(operation: (index: Int, Char, acc: Char) -> Char): Char {\n var index
= lastIndex\n if (index < 0) throw UnsupportedOperationException("Empty char sequence can't be reduced.")\n
var accumulator = get(index--)\n while (index >= 0) {\n accumulator = operation(index, get(index),
accumulator)\n --index\n }\n return accumulator}\n\n\n/**\n * Accumulates value starting with the last
character and applying [operation] from right to left\n * to each character with its index in the original char sequence
and current accumulator value.\n * \n * Returns `null` if the char sequence is empty.\n * \n * @param [operation]
function that takes the index of a character, the character itself and current accumulator value,\n * and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n\n@SinceKotlin("1.4")\npublic inline fun CharSequence.reduceRightIndexedOrNull(operation: (index: Int, Char,
acc: Char) -> Char): Char? {\n var index = lastIndex\n if (index < 0) return null\n var accumulator = get(index-
-)\n while (index >= 0) {\n accumulator = operation(index, get(index), accumulator)\n --index\n }\n
return accumulator}\n\n\n/**\n * Accumulates value starting with the last character and applying [operation] from
right to left\n * to each character and current accumulator value.\n * \n * Returns `null` if the char sequence is
empty.\n * \n * @param [operation] function that takes a character and current accumulator value,\n * and calculates

```

```

the next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun
CharSequence.reduceRightOrNull(operation: (Char, acc: Char) -> Char): Char? {\n    var index = lastIndex\n    if
(index < 0) return null\n    var accumulator = get(index--)\n    while (index >= 0) {\n        accumulator =
operation(get(index--), accumulator)\n    }\n    return accumulator\n}\n\n/**\n * Returns a list containing successive
accumulation values generated by applying [operation] from left to right\n * to each character and current
accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should
not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation]
function that takes current accumulator value and a character, and calculates the next accumulator value.\n * \n *
@sample samples.collections.Collections.Aggregates.runningFold\n
*\n@SinceKotlin("1.4")\npublic inline fun
<R> CharSequence.runningFold(initial: R, operation: (acc: R, Char) -> R): List<R> {\n    if (isEmpty()) return
listOf(initial)\n    val result = ArrayList<R>(length + 1).apply { add(initial) }\n    var accumulator = initial\n    for
(element in this) {\n        accumulator = operation(accumulator, element)\n        result.add(accumulator)\n    }\n    return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying
[operation] from left to right\n * to each character, its index in the original char sequence and current accumulator
value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be
mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that
takes the index of a character, current accumulator value\n * and the character itself, and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningFold\n
*\n@SinceKotlin("1.4")\npublic inline fun <R> CharSequence.runningFoldIndexed(initial: R, operation: (index:
Int, acc: R, Char) -> R): List<R> {\n    if (isEmpty()) return listOf(initial)\n    val result = ArrayList<R>(length +
1).apply { add(initial) }\n    var accumulator = initial\n    for (index in indices) {\n        accumulator =
operation(index, accumulator, this[index])\n        result.add(accumulator)\n    }\n    return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation] from left to right\n * to each character and current accumulator value that starts with the first character of this char sequence.\n * \n * Note
that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous
value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and a character,
and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningReduce\n
*\n@SinceKotlin("1.4")\npublic inline fun
CharSequence.runningReduce(operation: (acc: Char, Char) -> Char): List<Char> {\n    if (isEmpty()) return
emptyList()\n    var accumulator = this[0]\n    val result = ArrayList<Char>(length).apply { add(accumulator) }\n    for
(index in 1 until length) {\n        accumulator = operation(accumulator, this[index])\n        result.add(accumulator)\n    }\n    return result\n}\n\n/**\n * Returns a list containing successive accumulation
values generated by applying [operation] from left to right\n * to each character, its index in the original char
sequence and current accumulator value that starts with the first character of this char sequence.\n * \n * Note that
`acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in
resulting list.\n * \n * @param [operation] function that takes the index of a character, current accumulator value\n *
and the character itself, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningReduce\n
*\n@SinceKotlin("1.4")\npublic inline fun
CharSequence.runningReduceIndexed(operation: (index: Int, acc: Char, Char) -> Char): List<Char> {\n    if
(isEmpty()) return emptyList()\n    var accumulator = this[0]\n    val result = ArrayList<Char>(length).apply {
add(accumulator) }\n    for (index in 1 until length) {\n        accumulator = operation(index, accumulator,
this[index])\n        result.add(accumulator)\n    }\n    return result\n}\n\n/**\n * Returns a list containing successive
accumulation values generated by applying [operation] from left to right\n * to each character and current
accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should
not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation]
function that takes current accumulator value and a character, and calculates the next accumulator value.\n * \n *
@sample samples.collections.Collections.Aggregates.scan\n

```

```

*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun <R>
CharSequence.scan(initial: R, operation: (acc: R, Char) -> R): List<R> {\n  return runningFold(initial,
operation)\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying [operation]
from left to right\n * to each character, its index in the original char sequence and current accumulator value that
starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes the
index of a character, current accumulator value\n * and the character itself, and calculates the next accumulator
value.\n * \n * @sample samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic inline fun <R>
CharSequence.scanIndexed(initial: R, operation: (index: Int, acc: R, Char) -> R): List<R> {\n  return
runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each character in the char sequence.\n * \n * @Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun
CharSequence.sumBy(selector: (Char) -> Int): Int {\n  var sum: Int = 0\n  for (element in this) {\n    sum +=
selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each character in the char sequence.\n * \n * @Deprecated("Use sumOf instead.")
ReplaceWith("this.sumOf(selector)")\n@DeprecatedSinceKotlin(warningSince = "1.5")\npublic inline fun
CharSequence.sumByDouble(selector: (Char) -> Double): Double {\n  var sum: Double = 0.0\n  for (element in
this) {\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each character in the char sequence.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfDouble")\n@kotlin.internal.InlineOnly\npublic inline fun
CharSequence.sumOf(selector: (Char) -> Double): Double {\n  var sum: Double = 0.toDouble()\n  for (element in
this) {\n    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each character in the char sequence.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfInt")\n@kotlin.internal.InlineOnly\npublic inline fun
CharSequence.sumOf(selector: (Char) -> Int): Int {\n  var sum: Int = 0.toInt()\n  for (element in this) {\n   
sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each character in the char sequence.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfLong")\n@kotlin.internal.InlineOnly\npublic inline fun
CharSequence.sumOf(selector: (Char) -> Long): Long {\n  var sum: Long = 0.toLong()\n  for (element in this) {\n
    sum += selector(element)\n  }\n  return sum\n}\n\n/**\n * Returns the sum of all values produced by
[selector] function applied to each character in the char sequence.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfUInt")\n@WasExperimental(ExperimentalUnsignedType
s::class)\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.sumOf(selector: (Char) -> UInt): UInt {\n
var sum: UInt = 0.toUInt()\n  for (element in this) {\n    sum += selector(element)\n  }\n  return
sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each character in the char
sequence.\n
*\n@SinceKotlin("1.5")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@kotlin.jvm.JvmName("sumOfULong")\n@WasExperimental(ExperimentalUnsignedTy
pes::class)\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.sumOf(selector: (Char) -> ULong):
ULong {\n  var sum: ULong = 0.toULong()\n  for (element in this) {\n    sum += selector(element)\n  }\n
return sum\n}\n\n/**\n * Splits this char sequence into a list of strings each not exceeding the given [size].\n * \n *
The last string in the resulting list may have fewer characters than the given [size].\n * \n * @param size the number
of elements to take in each string, must be positive and can be greater than the number of elements in this char

```

```

sequence.\n * \n * @sample samples.text.Strings.chunked\n * \n * @SinceKotlin("1.2")\npublic fun
CharSequence.chunked(size: Int): List<String> {\n    return windowed(size, size, partialWindows =
true)\n}\n\n/**\n * Splits this char sequence into several char sequences each not exceeding the given [size]\n * and
applies the given [transform] function to an each.\n * \n * @return list of results of the [transform] applied to an
each char sequence.\n * \n * Note that the char sequence passed to the [transform] function is ephemeral and is valid
only inside that function.\n * You should not store it or allow it to escape in some way, unless you made a snapshot
of it.\n * The last char sequence may have fewer characters than the given [size].\n * \n * @param size the number
of elements to take in each char sequence, must be positive and can be greater than the number of elements in this
char sequence.\n * \n * @sample samples.text.Strings.chunkedTransform\n * \n * @SinceKotlin("1.2")\npublic fun
<R> CharSequence.chunked(size: Int, transform: (CharSequence) -> R): List<R> {\n    return windowed(size, size,
partialWindows = true, transform = transform)\n}\n\n/**\n * Splits this char sequence into a sequence of strings
each not exceeding the given [size].\n * \n * The last string in the resulting sequence may have fewer characters than
the given [size].\n * \n * @param size the number of elements to take in each string, must be positive and can be
greater than the number of elements in this char sequence.\n * \n * @sample
samples.collections.Collections.Transformations.chunked\n * \n * @SinceKotlin("1.2")\npublic fun
CharSequence.chunkedSequence(size: Int): Sequence<String> {\n    return chunkedSequence(size) { it.toString()
}\n}\n\n/**\n * Splits this char sequence into several char sequences each not exceeding the given [size]\n * and
applies the given [transform] function to an each.\n * \n * @return sequence of results of the [transform] applied to
an each char sequence.\n * \n * Note that the char sequence passed to the [transform] function is ephemeral and is
valid only inside that function.\n * You should not store it or allow it to escape in some way, unless you made a
snapshot of it.\n * The last char sequence may have fewer characters than the given [size].\n * \n * @param size the
number of elements to take in each char sequence, must be positive and can be greater than the number of elements
in this char sequence.\n * \n * @sample samples.text.Strings.chunkedTransformToSequence\n
*\n * @SinceKotlin("1.2")\npublic fun <R> CharSequence.chunkedSequence(size: Int, transform: (CharSequence) -
> R): Sequence<R> {\n    return windowedSequence(size, size, partialWindows = true, transform =
transform)\n}\n\n/**\n * Splits the original char sequence into pair of char sequences,\n * where *first* char
sequence contains characters for which [predicate] yielded `true`,\n * while *second* char sequence contains
characters for which [predicate] yielded `false`.\n * \n * @sample samples.text.Strings.partition\n * \n * @public inline
fun CharSequence.partition(predicate: (Char) -> Boolean): Pair<CharSequence, CharSequence> {\n    val first =
StringBuilder()\n    val second = StringBuilder()\n    for (element in this) {\n        if (predicate(element)) {\n
first.append(element)\n        } else {\n            second.append(element)\n        }\n    }\n    return Pair(first,
second)\n}\n\n/**\n * Splits the original string into pair of strings,\n * where *first* string contains characters for
which [predicate] yielded `true`,\n * while *second* string contains characters for which [predicate] yielded
`false`.\n * \n * @sample samples.text.Strings.partition\n * \n * @public inline fun String.partition(predicate: (Char) ->
Boolean): Pair<String, String> {\n    val first = StringBuilder()\n    val second = StringBuilder()\n    for (element in
this) {\n        if (predicate(element)) {\n            first.append(element)\n        } else {\n
second.append(element)\n        }\n    }\n    return Pair(first.toString(), second.toString())\n}\n\n/**\n * Returns a list
of snapshots of the window of the given [size]\n * sliding along this char sequence with the given [step], where
each\n * snapshot is a string.\n * \n * Several last strings may have fewer characters than the given [size].\n * \n *
Both [size] and [step] must be positive and can be greater than the number of elements in this char sequence.\n *
\n * @param size the number of elements to take in each window\n * @param step the number of elements to move the
window forward by on an each step, by default 1\n * @param partialWindows controls whether or not to keep
partial windows in the end if any,\n * by default `false` which means partial windows won't be preserved\n * \n *
\n * @sample samples.collections.Sequences.Transformations.takeWindows\n * \n * @SinceKotlin("1.2")\npublic fun
CharSequence.windowed(size: Int, step: Int = 1, partialWindows: Boolean = false): List<String> {\n    return
windowed(size, step, partialWindows) { it.toString() }\n}\n\n/**\n * Returns a list of results of applying the given
[transform] function to\n * an each char sequence representing a view over the window of the given [size]\n *
sliding along this char sequence with the given [step].\n * \n * Note that the char sequence passed to the [transform]

```

function is ephemeral and is valid only inside that function.

\* You should not store it or allow it to escape in some way, unless you made a snapshot of it.

\* Several last char sequences may have fewer characters than the given [size].

\* Both [size] and [step] must be positive and can be greater than the number of elements in this char sequence.

\* @param size the number of elements to take in each window

\* @param step the number of elements to move the window forward by on an each step, by default 1

\* @param partialWindows controls whether or not to keep partial windows in the end if any,

\* by default `false` which means partial windows won't be preserved

\* @sample samples.collections.Sequences.Transformations.averageWindows

```

*^@SinceKotlin("1.2")
public fun <R> CharSequence.windowed(size: Int, step: Int = 1, partialWindows: Boolean = false, transform: (CharSequence) -> R): List<R> {
    checkWindowSizeStep(size, step)
    val thisSize = this.length
    val resultCapacity = thisSize / step + if (thisSize % step == 0) 0 else 1
    val result = ArrayList<R>(resultCapacity)
    var index = 0
    while (index in 0 until thisSize) {
        val end = index + size
        val coercedEnd = if (end < 0 || end > thisSize) { if (partialWindows) thisSize else break } else end
        result.add(transform(subSequence(index, coercedEnd)))
        index += step
    }
    return result
}

```

\* Returns a sequence of snapshots of the window of the given [size] sliding along this char sequence with the given [step], where each snapshot is a string.

\* Several last strings may have fewer characters than the given [size].

\* Both [size] and [step] must be positive and can be greater than the number of elements in this char sequence.

\* @param size the number of elements to take in each window

\* @param step the number of elements to move the window forward by on an each step, by default 1

\* @param partialWindows controls whether or not to keep partial windows in the end if any,

\* by default `false` which means partial windows won't be preserved

\* @sample samples.collections.Sequences.Transformations.takeWindows

```

*^@SinceKotlin("1.2")
public fun CharSequence.windowedSequence(size: Int, step: Int = 1, partialWindows: Boolean = false): Sequence<String> {
    return windowedSequence(size, step, partialWindows) { it.toString() }
}

```

\* Returns a sequence of results of applying the given [transform] function to an each char sequence representing a view over the window of the given [size] sliding along this char sequence with the given [step].

\* Note that the char sequence passed to the [transform] function is ephemeral and is valid only inside that function.

\* You should not store it or allow it to escape in some way, unless you made a snapshot of it.

\* Several last char sequences may have fewer characters than the given [size].

\* Both [size] and [step] must be positive and can be greater than the number of elements in this char sequence.

\* @param size the number of elements to take in each window

\* @param step the number of elements to move the window forward by on an each step, by default 1

\* @param partialWindows controls whether or not to keep partial windows in the end if any,

\* by default `false` which means partial windows won't be preserved

\* @sample samples.collections.Sequences.Transformations.averageWindows

```

*^@SinceKotlin("1.2")
public fun <R> CharSequence.windowedSequence(size: Int, step: Int = 1, partialWindows: Boolean = false, transform: (CharSequence) -> R): Sequence<R> {
    checkWindowSizeStep(size, step)
    val windows = if (partialWindows) indices else 0 until length - size + 1
    step step
    return windows.asSequence().map { index ->
        val end = index + size
        val coercedEnd = if (end < 0 || end > length) length else end
        transform(subSequence(index, coercedEnd))
    }
}

```

\* Returns a list of pairs built from the characters of `this` and the [other] char sequences with the same index

\* The returned list has length of the shortest char sequence.

\* @sample samples.text.Strings.zip

```

*^public infix fun CharSequence.zip(other: CharSequence): List<Pair<Char, Char>> {
    return zip(other) { c1, c2 -> c1 to c2 }
}

```

\* Returns a list of values built from the characters of `this` and the [other] char sequences with the same index using the provided [transform] function applied to each pair of characters.

\* The returned list has length of the shortest char sequence.

\* @sample samples.text.Strings.zipWithTransform

```

*^public inline fun <V> CharSequence.zip(other: CharSequence, transform: (a: Char, b: Char) -> V): List<V> {
    val length = minOf(this.length, other.length)
    val list = ArrayList<V>(length)
    for (i in 0 until length) {
        list.add(transform(this[i], other[i]))
    }
    return list
}

```

\* Returns a list of pairs of each two adjacent characters in this char sequence.

\* The returned list is empty if this char sequence contains less than two characters.

\* @sample samples.collections.Collections.Transformations.zipWithNext

```

*^@SinceKotlin("1.2")
public fun

```

```

CharSequence.zipWithNext(): List<Pair<Char, Char>> {
    return zipWithNext { a, b -> a to b }
}
Returns a list containing the results of applying the given [transform] function to an each pair of two adjacent characters in this char sequence. The returned list is empty if this char sequence contains less than two characters.
@sample samples.collections.Collections.Transformations.zipWithNextToFindDeltas
Since Kotlin(1.2)
public inline fun <R> CharSequence.zipWithNext(transform: (a: Char, b: Char) -> R): List<R> {
    val size = length - 1
    if (size < 1) return emptyList()
    val result = ArrayList<R>(size)
    for (index in 0 until size) {
        result.add(transform(this[index], this[index + 1]))
    }
    return result
}
Creates an [Iterable] instance that wraps the original char sequence returning its characters when being iterated.
public fun CharSequence.asIterable(): Iterable<Char> {
    if (this is String && isEmpty()) return emptyList()
    return Iterable { this.iterator() }
}
Creates a [Sequence] instance that wraps the original char sequence returning its characters when being iterated.
public fun CharSequence.asSequence(): Sequence<Char> {
    if (this is String && isEmpty()) return emptySequence()
    return Sequence { this.iterator() }
}
Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors. Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.
@file:kotlin.jvm.JvmMultifileClass
@file:kotlin.jvm.JvmName("StringsKt")
package kotlin.text
import kotlin.contracts.contract
import kotlin.jvm.JvmName
Returns a copy of this string converted to upper case using the rules of the default locale.
@Deprecated("Use uppercase() instead.", ReplaceWith("uppercase()"))
@DeprecatedSinceKotlin(warningSince = "1.5")
public expect fun String.toUpperCase(): String
Returns a copy of this string converted to upper case using Unicode mapping rules of the invariant locale. This function supports one-to-many and many-to-one character mapping, thus the length of the returned string can be different from the length of the original string.
@sample samples.text.Strings.toUpperCase
Since Kotlin(1.5)
@WasExperimental(ExperimentalStdlibApi::class)
public expect fun String.toUpperCase(): String
Returns a copy of this string converted to lower case using the rules of the default locale.
@Deprecated("Use lowercase() instead.", ReplaceWith("lowercase()"))
@DeprecatedSinceKotlin(warningSince = "1.5")
public expect fun String.toLowerCase(): String
Returns a copy of this string converted to lower case using Unicode mapping rules of the invariant locale. This function supports one-to-many and many-to-one character mapping, thus the length of the returned string can be different from the length of the original string.
@sample samples.text.Strings.toLowerCase
Since Kotlin(1.5)
@WasExperimental(ExperimentalStdlibApi::class)
public expect fun String.toLowerCase(): String
Returns a copy of this string having its first letter titlecased using the rules of the default locale, or the original string if it's empty or already starts with a title case letter. The title case of a character is usually the same as its upper case with several exceptions. The particular list of characters with the special title case form depends on the underlying platform.
@sample samples.text.Strings.capitalize
@Deprecated("Use replaceFirstChar instead.", ReplaceWith("replaceFirstChar { if (it.isLowerCase()) it.titlecase() else it.toString() }"))
@DeprecatedSinceKotlin(warningSince = "1.5")
public expect fun String.capitalize(): String
Returns a copy of this string having its first letter lowercased using the rules of the default locale, or the original string if it's empty or already starts with a lower case letter.
@sample samples.text.Strings.decapitalize
@Deprecated("Use replaceFirstChar instead.", ReplaceWith("replaceFirstChar { it.lowercase() }"))
@DeprecatedSinceKotlin(warningSince = "1.5")
public expect fun String.decapitalize(): String
Returns a sub sequence of this char sequence having leading and trailing characters matching the [predicate] removed.
public inline fun CharSequence.trim(predicate: (Char) -> Boolean): CharSequence {
    var startIndex = 0
    var endIndex = length - 1
    var startFound = false
    while (startIndex <= endIndex) {
        val index = if (!startFound) startIndex else endIndex
        val match = predicate(this[index])
        if (!startFound) {
            if (!match) startFound = true
            else startIndex += 1
        } else {
            if (!match) break
            else endIndex -= 1
        }
    }
    return subSequence(startIndex, endIndex + 1)
}
Returns a string having leading and

```

trailing characters matching the [predicate] removed.

```

public inline fun String.trim(predicate: (Char) -> Boolean): String = (this as CharSequence).trim(predicate).toString()

```

\* Returns a sub sequence of this char sequence having leading characters matching the [predicate] removed.

```

public inline fun CharSequence.trimStart(predicate: (Char) -> Boolean): CharSequence {
    for (index in this.indices)
        if (!predicate(this[index]))
            return subSequence(index, length)
    return ""
}

```

\* Returns a string having leading characters matching the [predicate] removed.

```

public inline fun String.trimStart(predicate: (Char) -> Boolean): String = (this as CharSequence).trimStart(predicate).toString()

```

\* Returns a sub sequence of this char sequence having trailing characters matching the [predicate] removed.

```

public inline fun CharSequence.trimEnd(predicate: (Char) -> Boolean): CharSequence {
    for (index in this.indices.reversed())
        if (!predicate(this[index]))
            return subSequence(0, index + 1)
    return ""
}

```

\* Returns a string having trailing characters matching the [predicate] removed.

```

public inline fun String.trimEnd(predicate: (Char) -> Boolean): String = (this as CharSequence).trimEnd(predicate).toString()

```

\* Returns a sub sequence of this char sequence having leading and trailing characters from the [chars] array removed.

```

public fun CharSequence.trim(vararg chars: Char): CharSequence = trim { it in chars }

```

\* Returns a string having leading and trailing characters from the [chars] array removed.

```

public fun String.trim(vararg chars: Char): String = trim { it in chars }

```

\* Returns a sub sequence of this char sequence having leading characters from the [chars] array removed.

```

public fun CharSequence.trimStart(vararg chars: Char): CharSequence = trimStart { it in chars }

```

\* Returns a string having leading characters from the [chars] array removed.

```

public fun String.trimStart(vararg chars: Char): String = trimStart { it in chars }

```

\* Returns a sub sequence of this char sequence having trailing characters from the [chars] array removed.

```

public fun CharSequence.trimEnd(vararg chars: Char): CharSequence = trimEnd { it in chars }

```

\* Returns a string having trailing characters from the [chars] array removed.

```

public fun String.trimEnd(vararg chars: Char): String = trimEnd { it in chars }

```

\* Returns a sub sequence of this char sequence having leading and trailing whitespace removed.

```

public fun CharSequence.trim(): CharSequence = trim(Char::isWhitespace)

```

\* Returns a string having leading and trailing whitespace removed.

```

@kotlin.internal.InlineOnly
public inline fun String.trim(): String = (this as CharSequence).trim().toString()

```

\* Returns a sub sequence of this char sequence having leading whitespace removed.

```

public fun CharSequence.trimStart(): CharSequence = trimStart(Char::isWhitespace)

```

\* Returns a string having leading whitespace removed.

```

@kotlin.internal.InlineOnly
public inline fun String.trimStart(): String = (this as CharSequence).trimStart().toString()

```

\* Returns a sub sequence of this char sequence having trailing whitespace removed.

```

public fun CharSequence.trimEnd(): CharSequence = trimEnd(Char::isWhitespace)

```

\* Returns a string having trailing whitespace removed.

```

@kotlin.internal.InlineOnly
public inline fun String.trimEnd(): String = (this as CharSequence).trimEnd().toString()

```

\* Returns a char sequence with content of this char sequence padded at the beginning

\* to the specified [length] with the specified character or space.

\* @param length the desired string length.

\* @param padChar the character to pad string with, if it has length less than the [length] specified. Space is used by default.

\* @return Returns a char sequence of length at least [length] consisting of `this` char sequence prepended with [padChar] as many times

\* as are necessary to reach that length.

\* @sample samples.text.Strings.padStart

```

public fun CharSequence.padStart(length: Int, padChar: Char = ' '): CharSequence {
    if (length < 0)
        throw IllegalArgumentException("Desired length $length is less than zero.")
    if (length <= this.length)
        return this.subSequence(0, this.length)
    val sb =
        StringBuilder(length)
    for (i in 1..(length - this.length))
        sb.append(padChar)
    sb.append(this)
    return sb
}

```

\* Pads the string to the specified [length] at the beginning with the specified character or space.

\* @param length the desired string length.

\* @param padChar the character to pad string with, if it has length less than the [length] specified. Space is used by default.

\* @return Returns a string of length at least [length] consisting of `this` string prepended with [padChar] as many times

\* as are necessary to reach that length.

\* @sample samples.text.Strings.padStart

```

public fun String.padStart(length: Int, padChar: Char = ' '): String = (this as CharSequence).padStart(length, padChar).toString()

```

\* Returns a char sequence with content of this char sequence padded at the end

\* to the specified [length] with the specified character or space.

\* @param

```

length the desired string length.\n * @param padChar the character to pad string with, if it has length less than the
[length] specified. Space is used by default.\n * @return Returns a char sequence of length at least [length]
consisting of `this` char sequence appended with [padChar] as many times\n * as are necessary to reach that
length.\n * @sample samples.text.Strings.padEnd\n */\npublic fun CharSequence.padEnd(length: Int, padChar: Char
= ' '): CharSequence {\n    if (length < 0)\n        throw IllegalArgumentException("Desired length $length is less
than zero.")\n    if (length <= this.length)\n        return this.subSequence(0, this.length)\n    val sb =
StringBuilder(length)\n    sb.append(this)\n    for (i in 1..(length - this.length))\n        sb.append(padChar)\n    return
sb\n}\n\n/**\n * Pads the string to the specified [length] at the end with the specified character or space.\n *\n *
@param length the desired string length.\n * @param padChar the character to pad string with, if it has length less
than the [length] specified. Space is used by default.\n * @return Returns a string of length at least [length]
consisting of `this` string appended with [padChar] as many times\n * as are necessary to reach that length.\n *
@sample samples.text.Strings.padEnd\n */\npublic fun String.padEnd(length: Int, padChar: Char = ' '): String =\n(this as CharSequence).padEnd(length, padChar).toString()\n\n/**\n * Returns `true` if this nullable char sequence is
either `null` or empty.\n *\n * @sample samples.text.Strings.stringIsNullOrEmpty\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence?.isNullOrEmpty(): Boolean {\n    contract {\n        returns(false) implies (this@isNullOrEmpty != null)\n    }\n    return this == null || this.length == 0\n}\n\n/**\n *
Returns `true` if this char sequence is empty (contains no characters).\n *\n * @sample
samples.text.Strings.stringIsEmpty\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.isEmpty():
Boolean = length == 0\n\n/**\n * Returns `true` if this char sequence is not empty.\n *\n * @sample
samples.text.Strings.stringIsNotEmpty\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun
CharSequence.isNotEmpty(): Boolean = length > 0\n\n// implemented differently in JVM and JS\n//\npublic fun
String.isBlank(): Boolean = length() == 0 || all { it.isWhitespace() }\n\n/**\n * Returns `true` if this char sequence
is not empty and contains some characters except of whitespace characters.\n *\n * @sample
samples.text.Strings.stringIsNotBlank\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun
CharSequence.isNotBlank(): Boolean = !isBlank()\n\n/**\n * Returns `true` if this nullable char sequence is either
`null` or empty or consists solely of whitespace characters.\n *\n * @sample
samples.text.Strings.stringIsNullOrBlank\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun
CharSequence?.isNullOrBlank(): Boolean {\n    contract {\n        returns(false) implies (this@isNullOrBlank !=
null)\n    }\n    return this == null || this.isBlank()\n}\n\n/**\n * Iterator for characters of the given char sequence.\n
*\n * @public operator fun CharSequence.iterator(): CharIterator = object : CharIterator() {\n    private var index = 0\n\n
public override fun nextChar(): Char = get(index++)\n    public override fun hasNext(): Boolean = index <
length\n}\n\n/**\n * Returns the string if it is not `null`, or the empty string otherwise.\n *\n * @sample
samples.text.Strings.stringIfEmpty\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun String?.orEmpty(): String = this
?: ""\n\n/**\n * Returns this char sequence if it's not empty\n * or the result of calling [defaultValue] function if the char sequence is empty.\n
*\n * @sample samples.text.Strings.stringIfEmpty\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun <C, R> C.ifEmpty(defaultValue: () ->
R): R where C : CharSequence, C : R =\n    if (isEmpty()) defaultValue() else this\n\n/**\n * Returns this char
sequence if it is not empty and doesn't consist solely of whitespace characters,\n * or the result of calling
[defaultValue] function otherwise.\n *\n * @sample samples.text.Strings.stringIfNotBlank\n */\n\n@kotlin.internal.InlineOnly\npublic inline fun <C, R> C.ifBlank(defaultValue: () ->
R): R where C : CharSequence, C : R =\n    if (isBlank()) defaultValue() else this\n\n/**\n * Returns the range of valid
character indices for this char sequence.\n *\n * @public val CharSequence.indices: IntRange\n    get() = 0..length -
1\n\n/**\n * Returns the index of the last character in the char sequence or -1 if it is empty.\n *\n * @public val
CharSequence.lastIndex: Int\n    get() = this.length - 1\n\n/**\n * Returns `true` if this CharSequence has Unicode
surrogate pair at the specified [index].\n *\n * @public fun CharSequence.hasSurrogatePairAt(index: Int): Boolean {\n    return index in 0..length - 2\n        && this[index].isHighSurrogate()\n        && this[index +
1].isLowSurrogate()\n}\n\n/**\n * Returns a substring specified by the given [range] of indices.\n *\n * @public fun
String.substring(range: IntRange): String = substring(range.start, range.endInclusive + 1)\n\n/**\n * Returns a

```

```

subsequence of this char sequence specified by the given [range] of indices.\n */\npublic fun
CharSequence.subSequence(range: IntRange): CharSequence = subSequence(range.start, range.endInclusive +
1)\n\n/**\n * Returns a subsequence of this char sequence.\n *\n * This extension is chosen only for invocation with
old-named parameters.\n * Replace parameter names with the same as those of [CharSequence.subSequence].\n
*\n */\n@kotlin.internal.InlineOnly\n@Suppress("\u0027EXTENSION_SHADOWED_BY_MEMBER\u0027") // false
warning\n@Deprecated("\u0027Use parameters named startIndex and endIndex.\u0027", ReplaceWith("\u0027subSequence(startIndex
= start, endIndex = end)\u0027"))\npublic inline fun String.subSequence(start: Int, end: Int): CharSequence =
subSequence(start, end)\n\n/**\n * Returns a substring of chars from a range of this char sequence starting at the
[startIndex] and ending right before the [endIndex].\n *\n * @param startIndex the start index (inclusive).\n *
@param endIndex the end index (exclusive). If not specified, the length of the char sequence is used.\n
*\n */\n@kotlin.internal.InlineOnly\npublic inline fun CharSequence.substring(startIndex: Int, endIndex: Int = length):
String = subSequence(startIndex, endIndex).toString()\n\n/**\n * Returns a substring of chars at indices from the
specified [range] of this char sequence.\n */\npublic fun CharSequence.substring(range: IntRange): String =
subSequence(range.start, range.endInclusive + 1).toString()\n\n/**\n * Returns a substring before the first
occurrence of [delimiter].\n * If the string does not contain the delimiter, returns [missingDelimiterValue] which
defaults to the original string.\n */\npublic fun String.substringBefore(delimiter: Char, missingDelimiterValue:
String = this): String {\n    val index = indexOf(delimiter)\n    return if (index == -1) missingDelimiterValue else
substring(0, index)\n}\n\n/**\n * Returns a substring before the first occurrence of [delimiter].\n * If the string does
not contain the delimiter, returns [missingDelimiterValue] which defaults to the original string.\n */\npublic fun
String.substringBefore(delimiter: String, missingDelimiterValue: String = this): String {\n    val index =
indexOf(delimiter)\n    return if (index == -1) missingDelimiterValue else substring(0, index)\n}\n\n/**\n * Returns
a substring after the first occurrence of [delimiter].\n * If the string does not contain the delimiter, returns
[missingDelimiterValue] which defaults to the original string.\n */\npublic fun String.substringAfter(delimiter:
Char, missingDelimiterValue: String = this): String {\n    val index = indexOf(delimiter)\n    return if (index == -1)
missingDelimiterValue else substring(index + 1, length)\n}\n\n/**\n * Returns a substring after the first occurrence
of [delimiter].\n * If the string does not contain the delimiter, returns [missingDelimiterValue] which defaults to the
original string.\n */\npublic fun String.substringAfter(delimiter: String, missingDelimiterValue: String = this):
String {\n    val index = indexOf(delimiter)\n    return if (index == -1) missingDelimiterValue else substring(index +
delimiter.length, length)\n}\n\n/**\n * Returns a substring before the last occurrence of [delimiter].\n * If the string
does not contain the delimiter, returns [missingDelimiterValue] which defaults to the original string.\n */\npublic
fun String.substringBeforeLast(delimiter: Char, missingDelimiterValue: String = this): String {\n    val index =
lastIndexOf(delimiter)\n    return if (index == -1) missingDelimiterValue else substring(0, index)\n}\n\n/**\n *
Returns a substring before the last occurrence of [delimiter].\n * If the string does not contain the delimiter, returns
[missingDelimiterValue] which defaults to the original string.\n */\npublic fun String.substringBeforeLast(delimiter:
String, missingDelimiterValue: String = this): String {\n    val index = lastIndexOf(delimiter)\n    return if (index ==
-1) missingDelimiterValue else substring(0, index)\n}\n\n/**\n * Returns a substring after the last occurrence of
[delimiter].\n * If the string does not contain the delimiter, returns [missingDelimiterValue] which defaults to the
original string.\n */\npublic fun String.substringAfterLast(delimiter: Char, missingDelimiterValue: String = this):
String {\n    val index = lastIndexOf(delimiter)\n    return if (index == -1) missingDelimiterValue else
substring(index + 1, length)\n}\n\n/**\n * Returns a substring after the last occurrence of [delimiter].\n * If the
string does not contain the delimiter, returns [missingDelimiterValue] which defaults to the original string.\n
*\n */\npublic fun String.substringAfterLast(delimiter: String, missingDelimiterValue: String = this): String {\n
    val index = lastIndexOf(delimiter)\n    return if (index == -1) missingDelimiterValue else substring(index +
delimiter.length, length)\n}\n\n/**\n * Returns a char sequence with content of this char sequence where its part at
the given range\n * is replaced with the [replacement] char sequence.\n * @param startIndex the index of the first
character to be replaced.\n * @param endIndex the index of the first character after the replacement to keep in the
string.\n */\npublic fun CharSequence.replaceRange(startIndex: Int, endIndex: Int, replacement: CharSequence):
CharSequence {\n    if (endIndex < startIndex)\n        throw IndexOutOfBoundsException("\u0027End index ($endIndex)

```

```

is less than start index ($startIndex).")\n    val sb = StringBuilder()\n    sb.appendRange(this, 0, startIndex)\n    sb.append(replacement)\n    sb.appendRange(this, endIndex, length)\n    return sb\n}\n\n/**\n * Replaces the part of the string at the given range with the [replacement] char sequence.\n * @param startIndex the index of the first character to be replaced.\n * @param endIndex the index of the first character after the replacement to keep in the string.\n */\n@kotlin.internal.InlineOnly\npublic inline fun String.replaceRange(startIndex: Int, endIndex: Int, replacement: CharSequence): String =\n    (this as CharSequence).replaceRange(startIndex, endIndex, replacement).toString()\n\n/**\n * Returns a char sequence with content of this char sequence where its part at the given [range]\n * is replaced with the [replacement] char sequence.\n * @param range The end index of the [range] is included in the part to be replaced.\n */\npublic fun CharSequence.replaceRange(range: IntRange, replacement: CharSequence): CharSequence =\n    replaceRange(range.start, range.endInclusive + 1, replacement)\n\n/**\n * Replace the part of string at the given [range] with the [replacement] string.\n * @param range The end index of the [range] is included in the part to be replaced.\n */\n@kotlin.internal.InlineOnly\npublic inline fun String.replaceRange(range: IntRange, replacement: CharSequence): String =\n    (this as CharSequence).replaceRange(range, replacement).toString()\n\n/**\n * Returns a char sequence with content of this char sequence where its part at the given range is removed.\n * @param startIndex the index of the first character to be removed.\n * @param endIndex the index of the first character after the removed part to keep in the string.\n * @param [endIndex] is not included in the removed part.\n */\npublic fun CharSequence.removeRange(startIndex: Int, endIndex: Int): CharSequence {\n    if (endIndex < startIndex)\n        throw IndexOutOfBoundsException("End index ($endIndex) is less than start index ($startIndex).")\n    if (endIndex == startIndex)\n        return this.subSequence(0, length)\n    val sb = StringBuilder(length - (endIndex - startIndex))\n    sb.appendRange(this, 0, startIndex)\n    sb.appendRange(this, endIndex, length)\n    return sb\n}\n\n/**\n * Removes the part of a string at a given range.\n * @param startIndex the index of the first character to be removed.\n * @param endIndex the index of the first character after the removed part to keep in the string.\n * @param [endIndex] is not included in the removed part.\n */\n@kotlin.internal.InlineOnly\npublic inline fun String.removeRange(startIndex: Int, endIndex: Int): String =\n    (this as CharSequence).removeRange(startIndex, endIndex).toString()\n\n/**\n * Returns a char sequence with content of this char sequence where its part at the given [range] is removed.\n * @param range The end index of the [range] is included in the removed part.\n */\npublic fun CharSequence.removeRange(range: IntRange): CharSequence =\n    removeRange(range.start, range.endInclusive + 1)\n\n/**\n * Removes the part of a string at the given [range].\n * @param range The end index of the [range] is included in the removed part.\n */\n@kotlin.internal.InlineOnly\npublic inline fun String.removeRange(range: IntRange): String =\n    (this as CharSequence).removeRange(range).toString()\n\n/**\n * If this char sequence starts with the given [prefix], returns a new char sequence\n * with the prefix removed. Otherwise, returns a new char sequence with the same characters.\n */\npublic fun CharSequence.removePrefix(prefix: CharSequence): CharSequence {\n    if (startsWith(prefix))\n        return subSequence(prefix.length, length)\n    return subSequence(0, length)\n}\n\n/**\n * If this string starts with the given [prefix], returns a copy of this string\n * with the prefix removed. Otherwise, returns this string.\n */\npublic fun String.removePrefix(prefix: CharSequence): String {\n    if (startsWith(prefix))\n        return substring(prefix.length)\n    return this\n}\n\n/**\n * If this char sequence ends with the given [suffix], returns a new char sequence\n * with the suffix removed. Otherwise, returns a new char sequence with the same characters.\n */\npublic fun CharSequence.removeSuffix(suffix: CharSequence): CharSequence {\n    if (endsWith(suffix))\n        return subSequence(0, length - suffix.length)\n    return subSequence(0, length)\n}\n\n/**\n * If this string ends with the given [suffix], returns a copy of this string\n * with the suffix removed. Otherwise, returns this string.\n */\npublic fun String.removeSuffix(suffix: CharSequence): String {\n    if (endsWith(suffix))\n        return substring(0, length - suffix.length)\n    return this\n}\n\n/**\n * When this char sequence starts with the given [prefix] and ends with the given [suffix],\n * returns a new char sequence having both the given [prefix] and [suffix] removed.\n * Otherwise returns a new char sequence with the same characters.\n */\npublic fun CharSequence.removeSurrounding(prefix: CharSequence, suffix: CharSequence): CharSequence {\n    if ((length >= prefix.length + suffix.length) && startsWith(prefix) && endsWith(suffix))\n        return subSequence(prefix.length, length - suffix.length)\n    return subSequence(0, length)\n}\n\n/**\n * Removes

```

from a string both the given [prefix] and [suffix] if and only if `!string.startsWith(prefix) && !string.endsWith(suffix)`. Otherwise returns this string unchanged.

```

public fun String.removeSurrounding(prefix: CharSequence, suffix: CharSequence): String {
    if ((length >= prefix.length + suffix.length) &&
        startsWith(prefix) && endsWith(suffix)) {
        return substring(prefix.length, length - suffix.length)
    }
    return this
}

```

When this char sequence starts with and ends with the given [delimiter], returns a new char sequence having this [delimiter] removed both from the start and end. Otherwise returns a new char sequence with the same characters.

```

public fun CharSequence.removeSurrounding(delimiter: CharSequence):
CharSequence = removeSurrounding(delimiter, delimiter)

```

Removes the given [delimiter] string from both the start and the end of this string if and only if it starts with and ends with the [delimiter]. Otherwise returns this string unchanged.

```

public fun String.removeSurrounding(delimiter: CharSequence): String =
removeSurrounding(delimiter, delimiter)

```

Replace part of string before the first occurrence of given delimiter with the [replacement] string. If the string does not contain the delimiter, returns [missingDelimiterValue] which defaults to the original string.

```

public fun String.replaceBefore(delimiter: Char, replacement: String,
missingDelimiterValue: String = this): String {
    val index = indexOf(delimiter)
    return if (index == -1) missingDelimiterValue else
        replaceRange(0, index, replacement)
}

```

Replace part of string before the first occurrence of given delimiter with the [replacement] string. If the string does not contain the delimiter, returns [missingDelimiterValue] which defaults to the original string.

```

public fun String.replaceBefore(delimiter: String, replacement: String,
missingDelimiterValue: String = this): String {
    val index = indexOf(delimiter)
    return if (index == -1) missingDelimiterValue else
        replaceRange(0, index, replacement)
}

```

Replace part of string after the first occurrence of given delimiter with the [replacement] string. If the string does not contain the delimiter, returns [missingDelimiterValue] which defaults to the original string.

```

public fun String.replaceAfter(delimiter: Char, replacement: String,
missingDelimiterValue: String = this): String {
    val index = indexOf(delimiter)
    return if (index == -1) missingDelimiterValue else
        replaceRange(index + 1, length, replacement)
}

```

Replace part of string after the first occurrence of given delimiter with the [replacement] string. If the string does not contain the delimiter, returns [missingDelimiterValue] which defaults to the original string.

```

public fun String.replaceAfter(delimiter: String, replacement: String,
missingDelimiterValue: String = this): String {
    val index = indexOf(delimiter)
    return if (index == -1) missingDelimiterValue else
        replaceRange(index + delimiter.length, length, replacement)
}

```

Replace part of string after the last occurrence of given delimiter with the [replacement] string. If the string does not contain the delimiter, returns [missingDelimiterValue] which defaults to the original string.

```

public fun String.replaceAfterLast(delimiter: String, replacement: String,
missingDelimiterValue: String = this): String {
    val index = lastIndexOf(delimiter)
    return if (index == -1) missingDelimiterValue else
        replaceRange(index + delimiter.length, length, replacement)
}

```

Replace part of string after the last occurrence of given delimiter with the [replacement] string. If the string does not contain the delimiter, returns [missingDelimiterValue] which defaults to the original string.

```

public fun String.replaceAfterLast(delimiter: Char, replacement: String,
missingDelimiterValue: String = this): String {
    val index = lastIndexOf(delimiter)
    return if (index == -1) missingDelimiterValue else
        replaceRange(index + 1, length, replacement)
}

```

Replace part of string before the last occurrence of given delimiter with the [replacement] string. If the string does not contain the delimiter, returns [missingDelimiterValue] which defaults to the original string.

```

public fun String.replaceBeforeLast(delimiter: Char, replacement: String,
missingDelimiterValue: String = this): String {
    val index = lastIndexOf(delimiter)
    return if (index == -1) missingDelimiterValue else
        replaceRange(0, index, replacement)
}

```

Replace part of string before the last occurrence of given delimiter with the [replacement] string. If the string does not contain the delimiter, returns [missingDelimiterValue] which defaults to the original string.

```

public fun String.replaceBeforeLast(delimiter: String, replacement: String,
missingDelimiterValue: String = this): String {
    val index = lastIndexOf(delimiter)
    return if (index == -1)
        missingDelimiterValue else
        replaceRange(0, index, replacement)
}

```

public fun String.replace(oldChar: Char, newChar: Char, ignoreCase: Boolean): String // JVM- and JS-specific

```

public fun String.replace(oldValue: String, newValue: String, ignoreCase: Boolean): String // JVM- and JS-specific

```

Returns a new string obtained by

replacing each substring of this char sequence that matches the given regular expression `regex` with the given [replacement]. The [replacement] can consist of any combination of literal text and `$`-substitutions. To treat the replacement string literally escape it with the `[kotlin.text.Regex.Companion.escapeReplacement]` method.

```

*^@kotlin.internal.InlineOnly\npublic inline fun CharSequence.replace(regex: Regex, replacement: String): String
= regex.replace(this, replacement)\n\n**\n * Returns a new string obtained by replacing each substring of this char
sequence that matches the given regular expression regex with the result of the given function [transform] that takes
[MatchResult] and returns a string to be used as a replacement for that match.
*^@kotlin.internal.InlineOnly\npublic inline fun CharSequence.replace(regex: Regex, noinline transform:
(MatchResult) -> CharSequence): String =\n    regex.replace(this, transform)\n\n**\n * Replaces the first
occurrence of the given regular expression [regex] in this char sequence with specified [replacement] expression.
*\n * @param replacement A replacement expression that can include substitutions. See [Regex.replaceFirst] for
details.
*^@kotlin.internal.InlineOnly\npublic inline fun CharSequence.replaceFirst(regex: Regex, replacement:
String): String = regex.replaceFirst(this, replacement)\n\n**\n * Returns a copy of this string having its first
character replaced with the result of the specified [transform], or the original string if it's empty.
*\n * @param transform function that takes the first character and returns the result of the transform applied to the character.
*\n * @sample samples.text.Strings.replaceFirstChar
*^@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\n@OptIn(kotlin.experimental.Exper
imentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@JvmName("replaceFirstCharWithC
har")\n@kotlin.internal.InlineOnly\npublic inline fun String.replaceFirstChar(transform: (Char) -> Char): String {\n
return if (isEmpty()) transform(this[0]) + substring(1) else this\n}\n\n**\n * Returns a copy of this string
having its first character replaced with the result of the specified [transform], or the original string if it's empty.
*\n * @param transform function that takes the first character and returns the result of the transform applied to the
character.
*\n * @sample samples.text.Strings.replaceFirstChar
*^@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\n@OptIn(kotlin.experimental.Exper
imentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@JvmName("replaceFirstCharWithC
harSequence")\n@kotlin.internal.InlineOnly\npublic inline fun String.replaceFirstChar(transform: (Char) ->
CharSequence): String {\n    return if (isEmpty()) transform(this[0]).toString() + substring(1) else
this\n}\n\n**\n * Returns `true` if this char sequence matches the given regular expression.
*^@kotlin.internal.InlineOnly\npublic inline infix fun CharSequence.matches(regex: Regex): Boolean =
regex.matches(this)\n\n**\n * Implementation of [regionMatches] for CharSequences.
*^@kotlin.internal.InlineOnly\npublic inline fun CharSequence.regionMatchesImpl(thisOffset: Int, other: CharSequence, otherOffset: Int, length: Int, ignoreCase:
Boolean): Boolean {\n    if ((otherOffset < 0) || (thisOffset < 0) || (thisOffset > this.length - length) || (otherOffset >
other.length - length)) {\n        return false\n    }\n    for (index in 0 until length) {\n        if (!this[thisOffset +
index].equals(other[otherOffset + index], ignoreCase))\n            return false\n    }\n    return true\n}\n\n**\n * Returns `true` if this char sequence starts with the specified character.
*^@kotlin.internal.InlineOnly\npublic fun CharSequence.startsWith(char: Char, ignoreCase: Boolean = false): Boolean =\n    this.length > 0 &&
this[0].equals(char, ignoreCase)\n\n**\n * Returns `true` if this char sequence ends with the specified character.
*^@kotlin.internal.InlineOnly\npublic fun CharSequence.endsWith(char: Char, ignoreCase: Boolean = false): Boolean =\n    this.length > 0 &&
this[lastIndex].equals(char, ignoreCase)\n\n**\n * Returns `true` if this char sequence starts with the specified
prefix.
*^@kotlin.internal.InlineOnly\npublic fun CharSequence.startsWith(prefix: CharSequence, ignoreCase: Boolean = false): Boolean {\n    if
(!ignoreCase && this is String && prefix is String)\n        return this.startsWith(prefix)\n    else\n        return
regionMatchesImpl(0, prefix, 0, prefix.length, ignoreCase)\n}\n\n**\n * Returns `true` if a substring of this char
sequence starting at the specified offset [startIndex] starts with the specified prefix.
*^@kotlin.internal.InlineOnly\npublic fun CharSequence.startsWith(prefix: CharSequence, startIndex: Int, ignoreCase: Boolean = false): Boolean {\n    if
(!ignoreCase && this is String && prefix is String)\n        return this.startsWith(prefix, startIndex)\n    else\n        return
regionMatchesImpl(startIndex, prefix, 0, prefix.length, ignoreCase)\n}\n\n**\n * Returns `true` if this char
sequence ends with the specified suffix.
*^@kotlin.internal.InlineOnly\npublic fun CharSequence.endsWith(suffix: CharSequence,

```

```

ignoreCase: Boolean = false): Boolean {\n  if (!ignoreCase && this is String && suffix is String)\n    return\n    this.endsWith(suffix)\n  else\n    return regionMatchesImpl(length - suffix.length, suffix, 0, suffix.length,\n    ignoreCase)\n}\n\n// common prefix and suffix\n/**\n * Returns the longest string `prefix` such that this char\n * sequence and [other] char sequence both start with this prefix,\n * taking care not to split surrogate pairs.\n * If this\n * and [other] have no common prefix, returns the empty string.\n * @param ignoreCase `true` to ignore character\n * case when matching a character. By default `false`.\n * @sample samples.text.Strings.commonPrefixWith\n */\npublic fun CharSequence.commonPrefixWith(other: CharSequence, ignoreCase: Boolean = false): String {\n  val shortestLength = minOf(this.length, other.length)\n  var i = 0\n  while (i < shortestLength &&\n    this[i].equals(other[i], ignoreCase = ignoreCase)) {\n    i++\n  }\n  if (this.hasSurrogatePairAt(i - 1) ||\n    other.hasSurrogatePairAt(i - 1)) {\n    i--\n  }\n  return subSequence(0, i).toString()\n}\n\n/**\n * Returns the\n * longest string `suffix` such that this char sequence and [other] char sequence both end with this suffix,\n * taking\n * care not to split surrogate pairs.\n * If this\n * and [other] have no common suffix, returns the empty string.\n * @param ignoreCase `true` to ignore character\n * case when matching a character. By default `false`.\n * @sample\n * samples.text.Strings.commonSuffixWith\n */\npublic fun CharSequence.commonSuffixWith(other: CharSequence,\n  ignoreCase: Boolean = false): String {\n  val thisLength = this.length\n  val otherLength = other.length\n  val\n  shortestLength = minOf(thisLength, otherLength)\n  var i = 0\n  while (i < shortestLength && this[thisLength -\n  i - 1].equals(other[otherLength - i - 1], ignoreCase = ignoreCase)) {\n    i++\n  }\n  if\n  (this.hasSurrogatePairAt(thisLength - i - 1) || other.hasSurrogatePairAt(otherLength - i - 1)) {\n    i--\n  }\n  return subSequence(thisLength - i, thisLength).toString()\n}\n\n// indexOfAny()\n/**\n * Finds the index of the\n * first occurrence of any of the specified [chars] in this char sequence,\n * starting from the specified [startIndex] and\n * optionally ignoring the case.\n * @param ignoreCase `true` to ignore character\n * case when matching a character.\n * By default `false`.\n * @return An index of the first occurrence of matched character from [chars] or -1 if none of\n * [chars] are found.\n */\npublic fun CharSequence.indexOfAny(chars: CharArray, startIndex: Int = 0, ignoreCase:\n  Boolean = false): Int {\n  if (!ignoreCase && chars.size == 1 && this is String) {\n    val char = chars.single()\n    return nativeIndexOf(char, startIndex)\n  }\n  for (index in startIndex.coerceAtLeast(0)..lastIndex) {\n    val\n    charAtIndex = get(index)\n    if (chars.any { it.equals(charAtIndex, ignoreCase) })\n      return index\n  }\n  return -1\n}\n\n/**\n * Finds the index of the last occurrence of any of the specified [chars] in this char sequence,\n * starting from the specified [startIndex] and optionally ignoring the case.\n * @param startIndex The index of\n * character to start searching at. The search proceeds backward toward the beginning of the string.\n * @param\n * ignoreCase `true` to ignore character case when matching a character. By default `false`.\n * @return An index of\n * the last occurrence of matched character from [chars] or -1 if none of [chars] are found.\n */\npublic fun\n  CharSequence.lastIndexOfAny(chars: CharArray, startIndex: Int = lastIndex, ignoreCase: Boolean = false): Int {\n  if (!ignoreCase && chars.size == 1 && this is String) {\n    val char = chars.single()\n    return\n    nativeLastIndexOf(char, startIndex)\n  }\n  for (index in startIndex.coerceAtMost(lastIndex) downTo 0) {\n    val\n    charAtIndex = get(index)\n    if (chars.any { it.equals(charAtIndex, ignoreCase) })\n      return index\n  }\n  return -1\n}\n\nprivate fun CharSequence.indexOf(other: CharSequence, startIndex: Int, endIndex: Int,\n  ignoreCase: Boolean, last: Boolean = false): Int {\n  val indices = if (!last)\n    startIndex.coerceAtLeast(0)..endIndex.coerceAtMost(length)\n  else\n    startIndex.coerceAtMost(lastIndex)\n    downTo endIndex.coerceAtLeast(0)\n  if (this is String && other is String) { // smart cast\n    for (index in\n    indices) {\n      if (other.regionMatches(0, this, index, other.length, ignoreCase))\n        return index\n    }\n  } else {\n    for (index in indices) {\n      if (other.regionMatchesImpl(0, this, index, other.length,\n      ignoreCase))\n        return index\n    }\n  }\n  return -1\n}\n\nprivate fun CharSequence.findAnyOf(strings:\n  Collection<String>, startIndex: Int, ignoreCase: Boolean, last: Boolean): Pair<Int, String>? {\n  if (!ignoreCase\n  && strings.size == 1) {\n    val string = strings.single()\n    val index = if (!last) indexOf(string, startIndex) else\n    lastIndexOf(string, startIndex)\n    return if (index < 0) null else index to string\n  }\n  val indices = if (!last)\n    startIndex.coerceAtLeast(0)..length else startIndex.coerceAtMost(lastIndex) downTo 0\n  if (this is String) {\n    for (index in indices) {\n      val matchingString = strings.firstOrNull { it.regionMatches(0, this, index, it.length,\n      ignoreCase) }\n      if (matchingString != null)\n        return index to matchingString\n    }\n  } else {\n
```

```

    for (index in indices) {
        val matchingString = strings.firstOrNull { it.regionMatchesImpl(0, this, index,
it.length, ignoreCase) }
        if (matchingString != null) return index to matchingString
    }
    return null
}

/** Finds the first occurrence of any of the specified [strings] in this char sequence,
starting from the specified [startIndex] and optionally ignoring the case.
@param ignoreCase `true` to ignore character case when matching a string. By default `false`.
@return A pair of an index of the first occurrence of matched string from [strings] and the string matched or `null` if none of [strings] are found.
To avoid ambiguous results when strings in [strings] have characters in common, this method proceeds from the beginning to the end of this string, and finds at each position the first element in [strings] that matches this string at that position.

public fun CharSequence.findAnyOf(strings: Collection<String>, startIndex: Int = 0, ignoreCase: Boolean = false): Pair<Int, String>? =
    findAnyOf(strings, startIndex, ignoreCase, last = false)

/** Finds the last occurrence of any of the specified [strings] in this char sequence,
starting from the specified [startIndex] and optionally ignoring the case.
@param startIndex The index of character to start searching at. The search proceeds backward toward the beginning of the string.
@param ignoreCase `true` to ignore character case when matching a string. By default `false`.
@return A pair of an index of the last occurrence of matched string from [strings] and the string matched or `null` if none of [strings] are found.
To avoid ambiguous results when strings in [strings] have characters in common, this method proceeds from the end toward the beginning of this string, and finds at each position the first element in [strings] that matches this string at that position.

public fun CharSequence.findLastAnyOf(strings: Collection<String>, startIndex: Int = lastIndexOfAny(strings: Collection<String>, startIndex: Int = 0, ignoreCase: Boolean = false): Int =
    findAnyOf(strings, startIndex, ignoreCase, last = true)

/** Finds the index of the first occurrence of any of the specified [strings] in this char sequence,
starting from the specified [startIndex] and optionally ignoring the case.
@param ignoreCase `true` to ignore character case when matching a string. By default `false`.
@return An index of the first occurrence of matched string from [strings] or -1 if none of [strings] are found.
To avoid ambiguous results when strings in [strings] have characters in common, this method proceeds from the beginning to the end of this string, and finds at each position the first element in [strings] that matches this string at that position.

public fun CharSequence.indexOfAny(strings: Collection<String>, startIndex: Int = 0, ignoreCase: Boolean = false): Int =
    findAnyOf(strings, startIndex, ignoreCase, last = false)?.first ?: -1

/** Finds the index of the last occurrence of any of the specified [strings] in this char sequence,
starting from the specified [startIndex] and optionally ignoring the case.
@param startIndex The index of character to start searching at. The search proceeds backward toward the beginning of the string.
@param ignoreCase `true` to ignore character case when matching a string. By default `false`.
@return An index of the last occurrence of matched string from [strings] or -1 if none of [strings] are found.
To avoid ambiguous results when strings in [strings] have characters in common, this method proceeds from the end toward the beginning of this string, and finds at each position the first element in [strings] that matches this string at that position.

public fun CharSequence.lastIndexOfAny(strings: Collection<String>, startIndex: Int = lastIndexOfAny(strings: Collection<String>, startIndex: Int = lastIndexOf, ignoreCase: Boolean = false): Int =
    findAnyOf(strings, startIndex, ignoreCase, last = true)?.first ?: -1

/** Returns the index within this string of the first occurrence of the specified character, starting from the specified [startIndex].
@param ignoreCase `true` to ignore character case when matching a character. By default `false`.
@return An index of the first occurrence of [char] or -1 if none is found.

public fun CharSequence.indexOf(char: Char, startIndex: Int = 0, ignoreCase: Boolean = false): Int {
    return if (ignoreCase || this !is String)
        indexOfAny(charArrayOf(char), startIndex, ignoreCase)
    else
        nativeIndexOf(char, startIndex)
}

/** Returns the index within this char sequence of the first occurrence of the specified [string],
starting from the specified [startIndex].
@param ignoreCase `true` to ignore character case when matching a string. By default `false`.
@return An index of the first occurrence of [string] or -1 if none is found.
@sample samples.text.Strings.indexOf

public fun CharSequence.indexOf(string: String, startIndex: Int = 0, ignoreCase: Boolean = false): Int {
    return if (ignoreCase || this !is String)
        indexOf(string, startIndex, length, ignoreCase)
    else
        nativeIndexOf(string, startIndex)
}

/** Returns the index within this char sequence of the last occurrence of the specified character,
starting from the specified

```

```

[startIndex].\n * \n * @param startIndex The index of character to start searching at. The search proceeds backward
toward the beginning of the string.\n * @param ignoreCase `true` to ignore character case when matching a
character. By default `false`.\n * @return An index of the last occurrence of [char] or -1 if none is found.\n
*\npublic fun CharSequence.lastIndexOf(char: Char, startIndex: Int = lastIndex, ignoreCase: Boolean = false): Int
{\n    return if (ignoreCase || this !is String)\n        lastIndexOfAny(charArrayOf(char), startIndex, ignoreCase)\n
    else\n        nativeLastIndexOf(char, startIndex)\n}\n\n/**\n * Returns the index within this char sequence of the last
occurrence of the specified [string],\n * starting from the specified [startIndex].\n * \n * @param startIndex The
index of character to start searching at. The search proceeds backward toward the beginning of the string.\n *
@param ignoreCase `true` to ignore character case when matching a string. By default `false`.\n * @return An index
of the last occurrence of [string] or -1 if none is found.\n */\npublic fun CharSequence.lastIndexOf(string: String,
startIndex: Int = lastIndex, ignoreCase: Boolean = false): Int {\n    return if (ignoreCase || this !is String)\n
indexOf(string, startIndex, 0, ignoreCase, last = true)\n    else\n        nativeLastIndexOf(string,
startIndex)\n}\n\n/**\n * Returns `true` if this char sequence contains the specified [other] sequence of characters as
a substring.\n * \n * @param ignoreCase `true` to ignore character case when comparing strings. By default `false`.\n
*/\n\n@Suppress("INAPPLICABLE_OPERATOR_MODIFIER")\npublic operator fun
CharSequence.contains(other: CharSequence, ignoreCase: Boolean = false): Boolean =\n    if (other is String)\n
indexOf(other, ignoreCase = ignoreCase) >= 0\n    else\n        indexOf(other, 0, length, ignoreCase) >=
0\n}\n\n/**\n * Returns `true` if this char sequence contains the specified character [char].\n * \n * @param
ignoreCase `true` to ignore character case when comparing characters. By default `false`.\n
*/\n\n@Suppress("INAPPLICABLE_OPERATOR_MODIFIER")\npublic operator fun CharSequence.contains(char:
Char, ignoreCase: Boolean = false): Boolean =\n    indexOf(char, ignoreCase = ignoreCase) >= 0\n}\n\n/**\n * Returns
`true` if this char sequence contains at least one match of the specified regular expression [regex].\n
*/\n\n@kotlin.internal.InlineOnly\npublic inline operator fun CharSequence.contains(regex: Regex): Boolean =
regex.containsMatchIn(this)\n\n// rangesDelimitedBy\n\nprivate class DelimitedRangesSequence(\n    private
val input: CharSequence,\n    private val startIndex: Int,\n    private val limit: Int,\n    private val getNextMatch:
CharSequence.(current: Int) -> Pair<Int, Int>?(): Sequence<IntRange> {\n\n    override fun iterator():
Iterator<IntRange> = object : Iterator<IntRange> {\n        var nextState: Int = -1 // -1 for unknown, 0 for done, 1 for
continue\n        var currentStartIndex: Int = startIndex.coerceIn(0, input.length)\n        var nextSearchIndex: Int =
currentStartIndex\n        var nextItem: IntRange? = null\n        var counter: Int = 0\n\n        private fun calcNext() {\n
            if (nextSearchIndex < 0) {\n                nextState = 0\n                nextItem = null\n            } else {\n
                if (limit > 0 && ++counter >= limit || nextSearchIndex > input.length) {\n                    nextItem =
currentStartIndex..input.lastIndex\n                    nextSearchIndex = -1\n                } else {\n                    val match =
input.getNextMatch(nextSearchIndex)\n                    if (match == null) {\n                        nextItem =
currentStartIndex..input.lastIndex\n                        nextSearchIndex = -1\n                    } else {\n                        val
(index, length) = match\n                        nextItem = currentStartIndex until index\n                        currentStartIndex
= index + length\n                        nextSearchIndex = currentStartIndex + if (length == 0) 1 else 0\n                    }\n
                }\n                nextState = 1\n            }\n        }\n\n        override fun next(): IntRange {\n            if (nextState ==
-1)\n                calcNext()\n            if (nextState == 0)\n                throw NoSuchElementException()\n            val
result = nextItem as IntRange\n            // Clean next to avoid keeping reference on yielded instance\n            nextItem = null\n
            nextState = -1\n            return result\n        }\n\n        override fun hasNext(): Boolean {\n            if (nextState == -1)\n                calcNext()\n            return nextState == 1\n        }\n    }\n}\n\n/**\n * Returns a
sequence of index ranges of substrings in this char sequence around occurrences of the specified [delimiters].\n * \n *
@param delimiters One or more characters to be used as delimiters.\n * @param startIndex The index to start
searching delimiters from.\n * No range having its start value less than [startIndex] is returned.\n * [startIndex] is
coerced to be non-negative and not greater than length of this string.\n * @param ignoreCase `true` to ignore
character case when matching a delimiter. By default `false`.\n * @param limit The maximum number of substrings
to return. Zero by default means no limit is set.\n */\nprivate fun CharSequence.rangesDelimitedBy(delimiters:
CharArray, startIndex: Int = 0, ignoreCase: Boolean = false, limit: Int = 0): Sequence<IntRange> {\n

```

```

requireNonNegativeLimit(limit)\n\n return DelimitedRangesSequence(this, startIndex, limit, { currentIndex ->\n
  indexOfAny(delimiters, currentIndex, ignoreCase = ignoreCase).let { if (it < 0) null else it to 1 }\n
})\n\n}\n\n\n/**\n * Returns a sequence of index ranges of substrings in this char sequence around occurrences of the
specified [delimiters].\n *\n * @param delimiters One or more strings to be used as delimiters.\n * @param
startIndex The index to start searching delimiters from.\n * No range having its start value less than [startIndex] is
returned.\n * [startIndex] is coerced to be non-negative and not greater than length of this string.\n * @param
ignoreCase `true` to ignore character case when matching a delimiter. By default `false`.\n * @param limit The
maximum number of substrings to return. Zero by default means no limit is set.\n *\n * To avoid ambiguous results
when strings in [delimiters] have characters in common, this method proceeds from\n * the beginning to the end of
this string, and finds at each position the first element in [delimiters]\n * that matches this string at that position.\n
*/\nprivate fun CharSequence.rangesDelimitedBy(delimiters: Array<out String>, startIndex: Int = 0, ignoreCase:
Boolean = false, limit: Int = 0): Sequence<IntRange> {\n requireNonNegativeLimit(limit)\n val delimitersList =
delimiters.asList()\n\n return DelimitedRangesSequence(this, startIndex, limit, { currentIndex ->
findAnyOf(delimitersList, currentIndex, ignoreCase = ignoreCase, last = false)?.let { it.first to it.second.length }
})\n\n}\n\n\ninternal fun requireNonNegativeLimit(limit: Int) =\n require(limit >= 0) { \"Limit must be non-
negative, but was $limit\" }\n\n\n// split\n\n/**\n * Splits this char sequence to a sequence of strings around
occurrences of the specified [delimiters].\n *\n * @param delimiters One or more strings to be used as delimiters.\n
*\n * @param ignoreCase `true` to ignore character case when matching a delimiter. By default `false`.\n * @param
limit The maximum number of substrings to return. Zero by default means no limit is set.\n *\n * To avoid
ambiguous results when strings in [delimiters] have characters in common, this method proceeds from\n * the
beginning to the end of this string, and finds at each position the first element in [delimiters]\n * that matches this
string at that position.\n */\npublic fun CharSequence.splitToSequence(vararg delimiters: String, ignoreCase:
Boolean = false, limit: Int = 0): Sequence<String> =\n rangesDelimitedBy(delimiters, ignoreCase = ignoreCase,
limit = limit).map { substring(it) }\n\n\n/**\n * Splits this char sequence to a list of strings around occurrences of the
specified [delimiters].\n *\n * @param delimiters One or more strings to be used as delimiters.\n * @param
ignoreCase `true` to ignore character case when matching a delimiter. By default `false`.\n * @param limit The
maximum number of substrings to return. Zero by default means no limit is set.\n *\n * To avoid ambiguous results
when strings in [delimiters] have characters in common, this method proceeds from\n * the beginning to the end of
this string, and matches at each position the first element in [delimiters]\n * that is equal to a delimiter in this
instance at that position.\n */\npublic fun CharSequence.split(vararg delimiters: String, ignoreCase: Boolean = false,
limit: Int = 0): List<String> {\n if (delimiters.size == 1) {\n val delimiter = delimiters[0]\n if
(!delimiter.isEmpty()) {\n return split(delimiter, ignoreCase, limit)\n }\n }\n\n return
rangesDelimitedBy(delimiters, ignoreCase = ignoreCase, limit = limit).asIterable().map { substring(it) }\n\n\n\n/**\n * Splits this char sequence to a sequence of strings around occurrences of the specified [delimiters].\n *\n * @param
delimiters One or more characters to be used as delimiters.\n * @param ignoreCase `true` to ignore character case
when matching a delimiter. By default `false`.\n * @param limit The maximum number of substrings to return.\n
*/\npublic fun CharSequence.splitToSequence(vararg delimiters: Char, ignoreCase: Boolean = false, limit: Int = 0):
Sequence<String> =\n rangesDelimitedBy(delimiters, ignoreCase = ignoreCase, limit = limit).map { substring(it)
}\n\n\n/**\n * Splits this char sequence to a list of strings around occurrences of the specified [delimiters].\n *\n *
@param delimiters One or more characters to be used as delimiters.\n * @param ignoreCase `true` to ignore
character case when matching a delimiter. By default `false`.\n * @param limit The maximum number of substrings
to return.\n */\npublic fun CharSequence.split(vararg delimiters: Char, ignoreCase: Boolean = false, limit: Int = 0):
List<String> {\n if (delimiters.size == 1) {\n return split(delimiters[0].toString(), ignoreCase, limit)\n }\n\n
return rangesDelimitedBy(delimiters, ignoreCase = ignoreCase, limit = limit).asIterable().map { substring(it)
}\n\n\n\n/**\n * Splits this char sequence to a list of strings around occurrences of the specified [delimiter].\n * This
is specialized version of split which receives single non-empty delimiter and offers better performance\n *\n *
@param delimiter String used as delimiter\n * @param ignoreCase `true` to ignore character case when matching a
delimiter. By default `false`.\n * @param limit The maximum number of substrings to return.\n */\nprivate fun

```

```

CharSequence.split(delimiter: String, ignoreCase: Boolean, limit: Int): List<String> {
    requireNonNegativeLimit(limit)
    var currentOffset = 0
    var nextIndex = indexOf(delimiter, currentOffset, ignoreCase)
    if (nextIndex == -1 || limit == 1) {
        return listOf(this.toString())
    }
    val isLimited = limit > 0
    val result = ArrayList<String>(if (isLimited) limit.coerceAtMost(10) else 10)
    do {
        result.add(substring(currentOffset, nextIndex))
        currentOffset = nextIndex + delimiter.length // Do not search for next occurrence if we're reaching limit
        if (isLimited && result.size == limit - 1) break
        nextIndex = indexOf(delimiter, currentOffset, ignoreCase)
    } while (nextIndex != -1)
    result.add(substring(currentOffset, length))
    return result
}

/** Splits this char sequence to a list of strings around matches of the given regular expression.
 * @param limit Non-negative value specifying the maximum number of substrings to return.
 * Zero by default means no limit is set.
 */
@kotlin.internal.InlineOnly
public inline fun CharSequence.split(regex: Regex, limit: Int = 0): List<String> =
    regex.split(this, limit)

/** Splits this char sequence to a sequence of strings around matches of the given regular expression.
 * @param limit Non-negative value specifying the maximum number of substrings to return.
 * Zero by default means no limit is set.
 * @sample samples.text.Strings.splitToSequence
 */
@SinceKotlin("1.6")
@WasExperimental(ExperimentalStdlibApi::class)
@kotlin.internal.InlineOnly
public inline fun CharSequence.splitToSequence(regex: Regex, limit: Int = 0): Sequence<String> =
    regex.splitToSequence(this, limit)

/** Splits this char sequence to a sequence of lines delimited by any of the following character sequences: CRLF, LF or CR.
 * The lines returned do not include terminating line separators.
 */
public fun CharSequence.lineSequence(): Sequence<String> = splitToSequence("\\r\\n", "\\n", "\\r")

/** Splits this char sequence to a list of lines delimited by any of the following character sequences: CRLF, LF or CR.
 * The lines returned do not include terminating line separators.
 */
public fun CharSequence.lines(): List<String> = lineSequence().toList()

/** Returns `true` if the contents of this char sequence are equal to the contents of the specified [other],
 * i.e. both char sequences contain the same number of the same characters in the same order.
 * @sample samples.text.Strings.contentEquals
 */
@SinceKotlin("1.5")
public expect infix fun CharSequence?.contentEquals(other: CharSequence?): Boolean

/** Returns `true` if the contents of this char sequence are equal to the contents of the specified [other],
 * optionally ignoring case difference.
 * @param ignoreCase `true` to ignore character case when comparing contents.
 * @sample samples.text.Strings.contentEquals
 */
@SinceKotlin("1.5")
public expect fun CharSequence?.contentEquals(other: CharSequence?, ignoreCase: Boolean): Boolean

internal fun CharSequence?.contentEqualsIgnoreCaseImpl(other: CharSequence?): Boolean {
    if (this is String && other is String) {
        return this.equals(other, ignoreCase = true)
    }
    if (this === other) return true
    if (this == null || other == null || this.length != other.length) return false
    for (i in 0 until length) {
        if (!this[i].equals(other[i], ignoreCase = true)) {
            return false
        }
    }
    return true
}

internal fun CharSequence?.contentEqualsImpl(other: CharSequence?): Boolean {
    if (this is String && other is String) {
        return this == other
    }
    if (this === other) return true
    if (this == null || other == null || this.length != other.length) return false
    for (i in 0 until length) {
        if (this[i] != other[i]) {
            return false
        }
    }
    return true
}

/** Returns `true` if the content of this string is equal to the word "true", `false` if it is equal to "false",
 * and throws an exception otherwise.
 * There is also a lenient version of the function available on nullable String, [String?.toBoolean].
 * Note that this function is case-sensitive.
 * @sample samples.text.Strings.toBooleanStrict
 */
@SinceKotlin("1.5")
public fun String.toBooleanStrict(): Boolean = when (this) {
    "true" -> true
    "false" -> false
    else -> throw IllegalArgumentException("The string doesn't represent a boolean value: $this")
}

/** Returns `true` if the content of this string is equal to the word "true", `false` if it is equal to "false",
 * and `null` otherwise.
 * There is also a lenient version of the function available on nullable String, [String?.toBoolean].
 * Note that this function is case-sensitive.
 * @sample samples.text.Strings.toBooleanStrictOrNull
 */
@SinceKotlin("1.5")
public fun String.toBooleanStrictOrNull(): Boolean? = when (this) {
    "true" -> true
    "false" -> false
    else -> null
}

/* Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.
 * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.
 */

```

```

Auto-generated file. DO NOT EDIT!
package kotlin
import kotlin.jvm.*
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@JvmInline
public value class UByteArray
@PublishedApi
internal constructor(@PublishedApi internal val storage: ByteArray) :
Collection<UByte> {
    /** Creates a new array of the specified [size], with all elements initialized to zero. */
    public constructor(size: Int) : this(ByteArray(size))
    /** Returns the array element at the given [index]. This method can be called using the index operator. If the [index] is out of bounds of this array, throws an [IndexOutOfBoundsException] except in Kotlin/JS where the behavior is unspecified. */
    public operator fun get(index: Int): UByte = storage[index].toUByte()
    /** Sets the element at the given [index] to the given [value]. This method can be called using the index operator. If the [index] is out of bounds of this array, throws an [IndexOutOfBoundsException] except in Kotlin/JS where the behavior is unspecified. */
    public operator fun set(index: Int, value: UByte) {
        storage[index] = value.toByte()
    }
    /** Returns the number of elements in the array. */
    public override val size: Int get() = storage.size
    /** Creates an iterator over the elements of the array. */
    public override operator fun iterator(): kotlin.collections.Iterator<UByte> = Iterator(storage)
    @Suppress("DEPRECATION_ERROR")
    private class Iterator(private val array: ByteArray) : UByteIterator() {
        private var index = 0
        override fun hasNext() = index < array.size
        override fun nextUByte() = if (index < array.size) array[index++].toUByte() else throw NoSuchElementException(index.toString())
        override fun contains(element: UByte): Boolean {
            // TODO: Eliminate this check after KT-30016 gets fixed. // Currently JS BE does not generate special bridge method for this method.
            @Suppress("USELESS_CAST")
            if ((element as Any?) !is UByte) return false
            return storage.contains(element.toByte())
        }
        override fun containsAll(elements: Collection<UByte>): Boolean {
            return (elements as Collection<*>).all { it is UByte && storage.contains(it.toByte()) }
        }
        override fun isEmpty(): Boolean = this.storage.size == 0
    }
    /** Creates a new array of the specified [size], where each element is calculated by calling the specified [init] function. The function [init] is called for each array element sequentially starting from the first one. It should return the value for an array element given its index. */
    @SinceKotlin("1.3")
    @ExperimentalUnsignedTypes
    @kotlin.internal.InlineOnly
    public inline fun UByteArray(size: Int, init: (Int) -> UByte): UByteArray {
        return UByteArray(ByteArray(size) { index -> init(index).toByte() })
    }
    @SinceKotlin("1.3")
    @ExperimentalUnsignedTypes
    @kotlin.internal.InlineOnly
    public inline fun ubyteArrayOf(vararg elements: UByte): UByteArray = elements
}
/* Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors. Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.
Auto-generated file. DO NOT EDIT!
package kotlin
import kotlin.jvm.*
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@JvmInline
public value class UIntArray
@PublishedApi
internal constructor(@PublishedApi internal val storage: IntArray) :
Collection<UInt> {
    /** Creates a new array of the specified [size], with all elements initialized to zero. */
    public constructor(size: Int) : this(IntArray(size))
    /** Returns the array element at the given [index]. This method can be called using the index operator. If the [index] is out of bounds of this array, throws an [IndexOutOfBoundsException] except in Kotlin/JS where the behavior is unspecified. */
    public operator fun get(index: Int): UInt = storage[index].toInt()
    /** Sets the element at the given [index] to the given [value]. This method can be called using the index operator. If the [index] is out of bounds of this array, throws an [IndexOutOfBoundsException] except in Kotlin/JS where the behavior is unspecified. */
    public operator fun set(index: Int, value: UInt) {
        storage[index] = value.toInt()
    }
    /** Returns the number of elements in the array. */
    public override val size: Int get() = storage.size
    /** Creates an iterator over the elements of the array. */
    public override operator fun iterator(): kotlin.collections.Iterator<UInt> = Iterator(storage)
    @Suppress("DEPRECATION_ERROR")
    private class Iterator(private val array: IntArray) : UIntIterator() {
        private var index = 0
        override fun hasNext() = index < array.size
        override fun nextUInt() = if (index < array.size) array[index++].toInt() else throw NoSuchElementException(index.toString())
        override fun contains(element: UInt): Boolean {
            // TODO: Eliminate this check after KT-30016 gets fixed. // Currently JS BE does not generate special bridge method for this method.
            @Suppress("USELESS_CAST")
            if ((element as Any?) !is UInt) return false
            return storage.contains(element.toInt())
        }
        override fun containsAll(elements: Collection<UInt>): Boolean {
            return (elements as Collection<*>).all { it is UInt && storage.contains(it.toInt()) }
        }
        override fun isEmpty(): Boolean = this.storage.size == 0
    }
}

```

```

TODO: Eliminate this check after KT-30016 gets fixed.\n    // Currently JS BE does not generate special bridge
method for this method.\n    @Suppress(\\"USELESS_CAST\\")\n    if ((element as Any?) !is UInt) return
false\n\n    return storage.contains(element.toInt())\n    }\n\n    override fun containsAll(elements:
Collection<UInt>): Boolean {\n        return (elements as Collection<*>).all { it is UInt &&
storage.contains(it.toInt()) }\n    }\n\n    override fun isEmpty(): Boolean = this.storage.size == 0\n}\n\n/**\n * Creates a new array of the specified [size], where each element is calculated by calling the specified\n * [init]
function.\n * The function [init] is called for each array element sequentially starting from the first one.\n * It
should return the value for an array element given its index.\n
*\n*\n@SinceKotlin(\\"1.3\\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray(size: Int, init: (Int) -> UInt): UIntArray {\n    return UIntArray(IntArray(size) { index ->
init(index).toInt()
})\n}\n\n@SinceKotlin(\\"1.3\\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
uintArrayOf(vararg elements: UInt): UIntArray = elements\n", /*\n * Copyright 2010-2021 JetBrains s.r.o. and
Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n */\n\n// Auto-generated file. DO NOT EDIT!\n\npackage
kotlin\n\nimport kotlin.jvm.*\n\n@SinceKotlin(\\"1.3\\")\n@ExperimentalUnsignedTypes\n@JvmInline\npublic
value class UIntArray\n@PublishedApi\ninternal constructor(@PublishedApi internal val storage: LongArray) :
Collection<ULong> {\n\n    /** Creates a new array of the specified [size], with all elements initialized to zero. *\n
    public constructor(size: Int) : this(LongArray(size))\n\n    /**\n     * Returns the array element at the given [index].
This method can be called using the index operator.\n     * If the [index] is out of bounds of this array, throws
an [IndexOutOfBoundsException] except in Kotlin/JS\n     * where the behavior is unspecified.\n     *\n     public
operator fun get(index: Int): UInt = storage[index].toInt()\n\n     /**\n     * Sets the element at the given
[index] to the given [value]. This method can be called using the index operator.\n     * If the [index] is out of
bounds of this array, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n     * where the behavior is
unspecified.\n     *\n     public operator fun set(index: Int, value: UInt) {\n         storage[index] = value.toInt()\n
    }\n\n     /** Returns the number of elements in the array. *\n     public override val size: Int get() = storage.size\n
}\n\n     /** Creates an iterator over the elements of the array. *\n     public override operator fun iterator():
kotlin.collections.Iterator<ULong> = Iterator(storage)\n\n     @Suppress(\\"DEPRECATION_ERROR\\")\n     private
class Iterator(private val array: LongArray) : UIntIterator() {\n         private var index = 0\n         override fun
hasNext() = index < array.size\n         override fun nextUInt() = if (index < array.size) array[index++].toInt()
else throw NoSuchElementException(index.toString())\n     }\n\n     override fun contains(element: UInt): Boolean
{\n         // TODO: Eliminate this check after KT-30016 gets fixed.\n         // Currently JS BE does not generate
special bridge method for this method.\n         @Suppress(\\"USELESS_CAST\\")\n         if ((element as Any?) !is
UInt) return false\n         return storage.contains(element.toInt())\n     }\n\n     override fun
containsAll(elements: Collection<ULong>): Boolean {\n         return (elements as Collection<*>).all { it is UInt &&
storage.contains(it.toInt()) }\n     }\n\n     override fun isEmpty(): Boolean = this.storage.size == 0\n}\n\n/**\n * Creates a new array of the specified [size], where each element is calculated by calling the specified\n * [init]
function.\n * The function [init] is called for each array element sequentially starting from the first one.\n * It
should return the value for an array element given its index.\n
*\n*\n@SinceKotlin(\\"1.3\\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray(size: Int, init: (Int) -> UInt): UIntArray {\n    return UIntArray(LongArray(size) { index ->
init(index).toInt()
})\n}\n\n@SinceKotlin(\\"1.3\\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
uintArrayOf(vararg elements: UInt): UIntArray = elements\n", /*\n * Copyright 2010-2021 JetBrains s.r.o. and
Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n */\n\n// Auto-generated file. DO NOT EDIT!\n\npackage
kotlin\n\nimport kotlin.jvm.*\n\n@SinceKotlin(\\"1.3\\")\n@ExperimentalUnsignedTypes\n@JvmInline\npublic
value class UIntArray\n@PublishedApi\ninternal constructor(@PublishedApi internal val storage: ShortArray) :

```

```

Collection<UShort> {\n\n  /** Creates a new array of the specified [size], with all elements initialized to zero. *\n
  public constructor(size: Int) : this(ShortArray(size))\n\n  /**\n   * Returns the array element at the given [index].
  This method can be called using the index operator.\n   *\n   * If the [index] is out of bounds of this array, throws
  an [IndexOutOfBoundsException] except in Kotlin/JS\n   * where the behavior is unspecified.\n   *\n   public
  operator fun get(index: Int): UShort = storage[index].toUShort()\n\n  /**\n   * Sets the element at the given
  [index] to the given [value]. This method can be called using the index operator.\n   *\n   * If the [index] is out of
  bounds of this array, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n   * where the behavior is
  unspecified.\n   *\n   public operator fun set(index: Int, value: UShort) {\n    storage[index] = value.toShort()\n
  }\n\n  /** Returns the number of elements in the array. *\n   public override val size: Int get() = storage.size\n
  /** Creates an iterator over the elements of the array. *\n   public override operator fun iterator():
  kotlin.collections.Iterator<UShort> = Iterator(storage)\n\n  @Suppress("DEPRECATION_ERROR")\n  private
  class Iterator(private val array: ShortArray) : UShortIterator() {\n    private var index = 0\n    override fun
  hasNext() = index < array.size\n    override fun nextUShort() = if (index < array.size) array[index++].toUShort()
  else throw NoSuchElementException(index.toString())\n  }\n\n  override fun contains(element: UShort): Boolean
  {\n    // TODO: Eliminate this check after KT-30016 gets fixed.\n    // Currently JS BE does not generate
  special bridge method for this method.\n    @Suppress("USELESS_CAST")\n    if ((element as Any?) !is
  UShort) return false\n    return storage.contains(element.toShort())\n  }\n\n  override fun
  containsAll(elements: Collection<UShort>): Boolean {\n    return (elements as Collection<*>).all { it is UShort
  && storage.contains(it.toShort()) }\n  }\n\n  override fun isEmpty(): Boolean = this.storage.size == 0\n\n  /**\n
  * Creates a new array of the specified [size], where each element is calculated by calling the specified\n * [init]
  function.\n * \n * The function [init] is called for each array element sequentially starting from the first one.\n * It
  should return the value for an array element given its index.\n
  *\n  @SinceKotlin("1.3")\n  @ExperimentalUnsignedTypes\n  @kotlin.internal.InlineOnly\n  public inline fun
  UShortArray(size: Int, init: (Int) -> UShort): UShortArray {\n    return UShortArray(ShortArray(size) { index ->
  init(index).toShort()
  })\n}\n\n  @SinceKotlin("1.3")\n  @ExperimentalUnsignedTypes\n  @kotlin.internal.InlineOnly\n  public inline fun
  ushortArrayOf(vararg elements: UShort): UShortArray = elements\n  ,"/*\n * Copyright 2010-2021 JetBrains s.r.o.
  and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license
  that can be found in the license/LICENSE.txt file.\n
  *\n  @file:kotlin.jvm.JvmMultifileClass\n  @file:kotlin.jvm.JvmName("UArraysKt")\n  @file:kotlin.jvm.JvmPacka
  geName("kotlin.collections.unsigned")\n  @package kotlin.collections\n  @n// NOTE: THIS FILE IS AUTO-
  GENERATED by the GenerateStandardLib.kt\n  // See:
  https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n  @n\n  @nimport kotlin.random.*\n  @nimport
  kotlin.ranges.contains\n  @nimport kotlin.ranges.reversed\n  @n\n  /**\n   * Returns 1st *element* from the array.\n   * \n * If the
  size of this array is less than 1, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n   * where the behavior
  is unspecified.\n   *\n  @SinceKotlin("1.3")\n  @ExperimentalUnsignedTypes\n  @kotlin.internal.InlineOnly\n  public
  inline operator fun UIntArray.component1(): UInt {\n    return get(0)\n}\n\n  /**\n   * Returns 1st *element* from the
  array.\n   * \n * If the size of this array is less than 1, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n
  * where the behavior is unspecified.\n
  *\n  @SinceKotlin("1.3")\n  @ExperimentalUnsignedTypes\n  @kotlin.internal.InlineOnly\n  public inline operator fun
  ULongArray.component1(): ULong {\n    return get(0)\n}\n\n  /**\n   * Returns 1st *element* from the array.\n   * \n *
  If the size of this array is less than 1, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n   * where the
  behavior is unspecified.\n
  *\n  @SinceKotlin("1.3")\n  @ExperimentalUnsignedTypes\n  @kotlin.internal.InlineOnly\n  public inline operator fun
  UByteArray.component1(): UByte {\n    return get(0)\n}\n\n  /**\n   * Returns 1st *element* from the array.\n   * \n *
  If the size of this array is less than 1, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n   * where the
  behavior is unspecified.\n
  *\n  @SinceKotlin("1.3")\n  @ExperimentalUnsignedTypes\n  @kotlin.internal.InlineOnly\n  public inline operator fun

```

UShortArray.component1(): UShort { \n return get(0)\n}\n\n/\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \*  
If the size of this array is less than 2, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n \* where the behavior is unspecified.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun
```

UIntArray.component2(): UInt { \n return get(1)\n}\n\n/\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \*  
If the size of this array is less than 2, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n \* where the behavior is unspecified.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun
```

ULongArray.component2(): ULong { \n return get(1)\n}\n\n/\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \*  
If the size of this array is less than 2, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n \* where the behavior is unspecified.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun
```

UByteArray.component2(): UByte { \n return get(1)\n}\n\n/\*\*\n \* Returns 2nd \*element\* from the array.\n \* \n \*  
If the size of this array is less than 2, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n \* where the behavior is unspecified.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun
```

UShortArray.component2(): UShort { \n return get(1)\n}\n\n/\*\*\n \* Returns 3rd \*element\* from the array.\n \* \n \*  
If the size of this array is less than 3, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n \* where the behavior is unspecified.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun
```

UIntArray.component3(): UInt { \n return get(2)\n}\n\n/\*\*\n \* Returns 3rd \*element\* from the array.\n \* \n \*  
If the size of this array is less than 3, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n \* where the behavior is unspecified.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun
```

ULongArray.component3(): ULong { \n return get(2)\n}\n\n/\*\*\n \* Returns 3rd \*element\* from the array.\n \* \n \*  
If the size of this array is less than 3, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n \* where the behavior is unspecified.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun
```

UByteArray.component3(): UByte { \n return get(2)\n}\n\n/\*\*\n \* Returns 3rd \*element\* from the array.\n \* \n \*  
If the size of this array is less than 3, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n \* where the behavior is unspecified.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun
```

UShortArray.component3(): UShort { \n return get(2)\n}\n\n/\*\*\n \* Returns 4th \*element\* from the array.\n \* \n \*  
If the size of this array is less than 4, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n \* where the behavior is unspecified.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun
```

UIntArray.component4(): UInt { \n return get(3)\n}\n\n/\*\*\n \* Returns 4th \*element\* from the array.\n \* \n \*  
If the size of this array is less than 4, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n \* where the behavior is unspecified.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun
```

ULongArray.component4(): ULong { \n return get(3)\n}\n\n/\*\*\n \* Returns 4th \*element\* from the array.\n \* \n \*  
If the size of this array is less than 4, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n \* where the behavior is unspecified.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun
```

UByteArray.component4(): UByte { \n return get(3)\n}\n\n/\*\*\n \* Returns 4th \*element\* from the array.\n \* \n \*  
If the size of this array is less than 4, throws an [IndexOutOfBoundsException] except in Kotlin/JS\n \* where the behavior is unspecified.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun
```

UShortArray.component4(): UShort { \n return get(3)\n}\n\n/\*\*\n \* Returns 5th \*element\* from the array.\n \* \n \*

If the size of this array is less than 5, throws an [IndexOutOfBoundsException] except in Kotlin/JS where the behavior is unspecified.

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun  
UIntArray.component5(): UInt {\n    return get(4)\n}\n\n/**\n * Returns 5th *element* from the array.\n * \n * If the size of this array is less than 5, throws an [IndexOutOfBoundsException] except in Kotlin/JS where the behavior is unspecified.\n */\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun  
ULongArray.component5(): ULong {\n    return get(4)\n}\n\n/**\n * Returns 5th *element* from the array.\n * \n * If the size of this array is less than 5, throws an [IndexOutOfBoundsException] except in Kotlin/JS where the behavior is unspecified.\n */
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun  
UByteArray.component5(): UByte {\n    return get(4)\n}\n\n/**\n * Returns 5th *element* from the array.\n * \n * If the size of this array is less than 5, throws an [IndexOutOfBoundsException] except in Kotlin/JS where the behavior is unspecified.\n */
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline operator fun  
UShortArray.component5(): UShort {\n    return get(4)\n}\n\n/**\n * Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.elementAt
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic expect fun UIntArray.elementAt(index: Int):  
UInt\n\n/**\n * Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.elementAt
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic expect fun ULongArray.elementAt(index: Int):  
ULong\n\n/**\n * Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.elementAt
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic expect fun UByteArray.elementAt(index: Int):  
UByte\n\n/**\n * Returns an element at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.elementAt
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic expect fun UShortArray.elementAt(index: Int):  
UShort\n\n/**\n * Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.elementAtOrElse
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun  
UIntArray.elementAtOrElse(index: Int, defaultValue: (Int) -> UInt): UInt {\n    return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.elementAtOrElse
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun  
ULongArray.elementAtOrElse(index: Int, defaultValue: (Int) -> ULong): ULong {\n    return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.elementAtOrElse
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun  
UByteArray.elementAtOrElse(index: Int, defaultValue: (Int) -> UByte): UByte {\n    return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or the result of calling the [defaultValue] function if the [index] is out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.elementAtOrElse
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun  
UShortArray.elementAtOrElse(index: Int, defaultValue: (Int) -> UShort): UShort {\n    return if (index >= 0 && index <= lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or
```

```

`null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.elementAtOrNull(index: Int): UInt? {\n    return this.getOrNull(index)\n}\n\n/**\n * Returns an element
at the given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.elementAtOrNull(index: Int): ULong? {\n    return this.getOrNull(index)\n}\n\n/**\n * Returns an
element at the given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.elementAtOrNull(index: Int): UByte? {\n    return this.getOrNull(index)\n}\n\n/**\n * Returns an
element at the given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.elementAtOrNull\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.elementAtOrNull(index: Int): UShort? {\n    return this.getOrNull(index)\n}\n\n/**\n * Returns the
first element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.find(predicate: (UInt) -> Boolean): UInt? {\n    return firstOrNull(predicate)\n}\n\n/**\n * Returns the
first element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.find(predicate: (ULong) -> Boolean): ULong? {\n    return firstOrNull(predicate)\n}\n\n/**\n *
Returns the first element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.find(predicate: (UByte) -> Boolean): UByte? {\n    return firstOrNull(predicate)\n}\n\n/**\n * Returns
the first element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.find(predicate: (UShort) -> Boolean): UShort? {\n    return firstOrNull(predicate)\n}\n\n/**\n *
Returns the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.findLast(predicate: (UInt) -> Boolean): UInt? {\n    return lastOrNull(predicate)\n}\n\n/**\n * Returns
the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.findLast(predicate: (ULong) -> Boolean): ULong? {\n    return lastOrNull(predicate)\n}\n\n/**\n *
Returns the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.findLast(predicate: (UByte) -> Boolean): UByte? {\n    return lastOrNull(predicate)\n}\n\n/**\n *
Returns the last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.find\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.findLast(predicate: (UShort) -> Boolean): UShort? {\n    return lastOrNull(predicate)\n}\n\n/**\n *

```

Returns first element.\n \* @throws [NoSuchElementException] if the array is empty.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.first(): UInt {\n    return storage.first().toUInt()\n}\n\n/**\n * Returns first element.\n * @throws
[NoSuchElementException] if the array is empty.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.first(): ULong {\n    return storage.first().toULong()\n}\n\n/**\n * Returns first element.\n * @throws
[NoSuchElementException] if the array is empty.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.first(): UByte {\n    return storage.first().toUByte()\n}\n\n/**\n * Returns first element.\n * @throws
[NoSuchElementException] if the array is empty.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.first(): UShort {\n    return storage.first().toUShort()\n}\n\n/**\n * Returns the first element matching
the given [predicate].\n * @throws [NoSuchElementException] if no such element is found.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.first(predicate: (UInt) -> Boolean): UInt {\n    for (element in this) if (predicate(element)) return
element\n    throw NoSuchElementException("Array contains no element matching the predicate.")\n}\n\n/**\n *
Returns the first element matching the given [predicate].\n * @throws [NoSuchElementException] if no such
element is found.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.first(predicate: (ULong) -> Boolean): ULong {\n    for (element in this) if (predicate(element)) return
element\n    throw NoSuchElementException("Array contains no element matching the predicate.")\n}\n\n/**\n *
Returns the first element matching the given [predicate].\n * @throws [NoSuchElementException] if no such
element is found.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.first(predicate: (UByte) -> Boolean): UByte {\n    for (element in this) if (predicate(element)) return
element\n    throw NoSuchElementException("Array contains no element matching the predicate.")\n}\n\n/**\n *
Returns the first element matching the given [predicate].\n * @throws [NoSuchElementException] if no such
element is found.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.first(predicate: (UShort) -> Boolean): UShort {\n    for (element in this) if (predicate(element)) return
element\n    throw NoSuchElementException("Array contains no element matching the predicate.")\n}\n\n/**\n *
Returns the first element, or `null` if the array is empty.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.firstOrNull(): UInt? {\n    return
if (isEmpty()) null else this[0]\n}\n\n/**\n * Returns the first element, or `null` if the array is empty.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.firstOrNull(): ULong? {\n    return
if (isEmpty()) null else this[0]\n}\n\n/**\n * Returns the first element, or `null` if the array is empty.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.firstOrNull(): UByte? {\n    return
if (isEmpty()) null else this[0]\n}\n\n/**\n * Returns the first element, or `null` if the array is empty.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.firstOrNull(): UShort? {\n    return
if (isEmpty()) null else this[0]\n}\n\n/**\n * Returns the first element matching the given [predicate], or `null`
if element was not found.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.firstOrNull(predicate: (UInt) -> Boolean): UInt? {\n    for (element in this) if (predicate(element)) return
element\n    return null\n}\n\n/**\n * Returns the first element matching the given [predicate], or `null` if element
was not found.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic
inline fun ULongArray.firstOrNull(predicate: (ULong) -> Boolean): ULong? {\n    for (element in this) if
(predicate(element)) return element\n    return null\n}\n\n/**\n * Returns the first element matching the given
[predicate], or `null` if element was not found.\n
*\n
```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.firstOrNull(predicate: (UByte) -> Boolean): UByte? {\n  for (element in this) if (predicate(element))
return element\n  return null\n}\n\n/**\n * Returns the first element matching the given [predicate], or `null` if
element was not found.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.firstOrNull(predicate: (UShort) -> Boolean): UShort? {\n  for (element in this) if (predicate(element))
return element\n  return null\n}\n\n/**\n * Returns an element at the given [index] or the result of calling the
[defaultValue] function if the [index] is out of bounds of this array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.getOrElse(index: Int, defaultValue: (Int) -> UInt): UInt {\n  return if (index >= 0 && index <=
lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or the result of
calling the [defaultValue] function if the [index] is out of bounds of this array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.getOrElse(index: Int, defaultValue: (Int) -> ULong): ULong {\n  return if (index >= 0 && index <=
lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or the result of
calling the [defaultValue] function if the [index] is out of bounds of this array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.getOrElse(index: Int, defaultValue: (Int) -> UByte): UByte {\n  return if (index >= 0 && index <=
lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or the result of
calling the [defaultValue] function if the [index] is out of bounds of this array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.getOrElse(index: Int, defaultValue: (Int) -> UShort): UShort {\n  return if (index >= 0 && index <=
lastIndex) get(index) else defaultValue(index)\n}\n\n/**\n * Returns an element at the given [index] or `null` if the
[index] is out of bounds of this array.\n * \n * @sample samples.collections.Collections.Elements.getOrNull\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.getOrNull(index: Int): UInt? {\n
return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns an element at the given
[index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.getOrNull\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.getOrNull(index: Int):
ULong? {\n  return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns an element at
the given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.getOrNull\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.getOrNull(index: Int): UByte?
{\n  return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns an element at the
given [index] or `null` if the [index] is out of bounds of this array.\n * \n * @sample
samples.collections.Collections.Elements.getOrNull\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.getOrNull(index: Int):
UShort? {\n  return if (index >= 0 && index <= lastIndex) get(index) else null\n}\n\n/**\n * Returns first index of
[element], or -1 if the array does not contain element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.indexOf(element: UInt): Int {\n  return storage.indexOf(element.toInt())\n}\n\n/**\n * Returns first
index of [element], or -1 if the array does not contain element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.indexOf(element: ULong): Int {\n  return storage.indexOf(element.toLong())\n}\n\n/**\n * Returns
first index of [element], or -1 if the array does not contain element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.indexOf(element: UByte): Int {\n  return storage.indexOf(element.toByte())\n}\n\n/**\n * Returns
first index of [element], or -1 if the array does not contain element.\n

```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.indexOf(element: UShort): Int {\n    return storage.indexOf(element.toShort())\n}\n\n/**\n * Returns
index of the first element matching the given [predicate], or -1 if the array does not contain such element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.indexOfFirst(predicate: (UInt) -> Boolean): Int {\n    return storage.indexOfFirst { predicate(it.toUInt())
}\n}\n\n/**\n * Returns index of the first element matching the given [predicate], or -1 if the array does not contain
such element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic
inline fun ULongArray.indexOfFirst(predicate: (ULong) -> Boolean): Int {\n    return storage.indexOfFirst {
predicate(it.toULong()) }\n}\n\n/**\n * Returns index of the first element matching the given [predicate], or -1 if the
array does not contain such element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.indexOfFirst(predicate: (UByte) -> Boolean): Int {\n    return storage.indexOfFirst {
predicate(it.toUByte()) }\n}\n\n/**\n * Returns index of the first element matching the given [predicate], or -1 if the
array does not contain such element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.indexOfFirst(predicate: (UShort) -> Boolean): Int {\n    return storage.indexOfFirst {
predicate(it.toUShort()) }\n}\n\n/**\n * Returns index of the last element matching the given [predicate], or -1 if the
array does not contain such element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.indexOfLast(predicate: (UInt) -> Boolean): Int {\n    return storage.indexOfLast { predicate(it.toUInt())
}\n}\n\n/**\n * Returns index of the last element matching the given [predicate], or -1 if the array does not contain
such element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic
inline fun ULongArray.indexOfLast(predicate: (ULong) -> Boolean): Int {\n    return storage.indexOfLast {
predicate(it.toULong()) }\n}\n\n/**\n * Returns index of the last element matching the given [predicate], or -1 if the
array does not contain such element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.indexOfLast(predicate: (UByte) -> Boolean): Int {\n    return storage.indexOfLast {
predicate(it.toUByte()) }\n}\n\n/**\n * Returns index of the last element matching the given [predicate], or -1 if the
array does not contain such element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.indexOfLast(predicate: (UShort) -> Boolean): Int {\n    return storage.indexOfLast {
predicate(it.toUShort()) }\n}\n\n/**\n * Returns the last element.\n * \n * @throws NoSuchElementException if the
array is empty.\n * \n * @sample samples.collections.Collections.Elements.last\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.last(): UInt {\n    return storage.last().toUInt()\n}\n\n/**\n * Returns the last element.\n * \n * @throws
NoSuchElementException if the array is empty.\n * \n * @sample samples.collections.Collections.Elements.last\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.last(): ULong {\n    return storage.last().toULong()\n}\n\n/**\n * Returns the last element.\n * \n *
@throws NoSuchElementException if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.last(): UByte {\n    return storage.last().toUByte()\n}\n\n/**\n * Returns the last element.\n * \n *
@throws NoSuchElementException if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.last(): UShort {\n    return storage.last().toUShort()\n}\n\n/**\n * Returns the last element matching
the given [predicate].\n * \n * @throws NoSuchElementException if no such element is found.\n * \n * @sample
samples.collections.Collections.Elements.last\n

```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.last(predicate: (UInt) -> Boolean): UInt {\n  for (index in this.indices.reversed()) {\n    val element =
this[index]\n    if (predicate(element)) return element\n  }\n  throw NoSuchElementException("Array contains
no element matching the predicate.")\n}\n\n/**\n * Returns the last element matching the given [predicate].\n * \n *
@throws NoSuchElementException if no such element is found.\n * \n * @sample
samples.collections.Collections.Elements.last\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.last(predicate: (ULong) -> Boolean): ULong {\n  for (index in this.indices.reversed()) {\n    val
element = this[index]\n    if (predicate(element)) return element\n  }\n  throw
NoSuchElementException("Array contains no element matching the predicate.")\n}\n\n/**\n * Returns the last
element matching the given [predicate].\n * \n * @throws NoSuchElementException if no such element is found.\n
*\n * @sample samples.collections.Collections.Elements.last\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.last(predicate: (UByte) -> Boolean): UByte {\n  for (index in this.indices.reversed()) {\n    val
element = this[index]\n    if (predicate(element)) return element\n  }\n  throw
NoSuchElementException("Array contains no element matching the predicate.")\n}\n\n/**\n * Returns the last
element matching the given [predicate].\n * \n * @throws NoSuchElementException if no such element is found.\n
*\n * @sample samples.collections.Collections.Elements.last\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.last(predicate: (UShort) -> Boolean): UShort {\n  for (index in this.indices.reversed()) {\n    val
element = this[index]\n    if (predicate(element)) return element\n  }\n  throw
NoSuchElementException("Array contains no element matching the predicate.")\n}\n\n/**\n * Returns last index
of [element], or -1 if the array does not contain element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.lastIndexOf(element: UInt): Int {\n  return storage.lastIndexOf(element.toInt())\n}\n\n/**\n * Returns
last index of [element], or -1 if the array does not contain element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.lastIndexOf(element: ULong): Int {\n  return storage.lastIndexOf(element.toLong())\n}\n\n/**\n *
Returns last index of [element], or -1 if the array does not contain element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.lastIndexOf(element: UByte): Int {\n  return storage.lastIndexOf(element.toByte())\n}\n\n/**\n *
Returns last index of [element], or -1 if the array does not contain element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.lastIndexOf(element: UShort): Int {\n  return storage.lastIndexOf(element.toShort())\n}\n\n/**\n *
Returns the last element, or `null` if the array is empty.\n * \n * @sample
samples.collections.Collections.Elements.last\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic
fun UIntArray.lastOrNull(): UInt? {\n  return if (isEmpty()) null else this[size - 1]\n}\n\n/**\n * Returns the last
element, or `null` if the array is empty.\n * \n * @sample samples.collections.Collections.Elements.last\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.lastOrNull(): ULong? {\n
return if (isEmpty()) null else this[size - 1]\n}\n\n/**\n * Returns the last element, or `null` if the array is empty.\n
*\n * @sample samples.collections.Collections.Elements.last\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.lastOrNull(): UByte? {\n
return if (isEmpty()) null else this[size - 1]\n}\n\n/**\n * Returns the last element, or `null` if the array is empty.\n
*\n * @sample samples.collections.Collections.Elements.last\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.lastOrNull(): UShort? {\n
return if (isEmpty()) null else this[size - 1]\n}\n\n/**\n * Returns the last element matching the given [predicate], or
`null` if no such element was found.\n * \n * @sample samples.collections.Collections.Elements.last\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun

```

```

UIntArray.lastOrNull(predicate: (UInt) -> Boolean): UInt? {\n  for (index in this.indices.reversed()) {\n    val
element = this[index]\n    if (predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns the last
element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.lastOrNull(predicate: (ULong) -> Boolean): ULong? {\n  for (index in this.indices.reversed()) {\n
val element = this[index]\n    if (predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns the
last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.lastOrNull(predicate: (UByte) -> Boolean): UByte? {\n  for (index in this.indices.reversed()) {\n
val element = this[index]\n    if (predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns the
last element matching the given [predicate], or `null` if no such element was found.\n * \n * @sample
samples.collections.Collections.Elements.last\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.lastOrNull(predicate: (UShort) -> Boolean): UShort? {\n  for (index in this.indices.reversed()) {\n
val element = this[index]\n    if (predicate(element)) return element\n  }\n  return null\n}\n\n/**\n * Returns a
random element from this array.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.random(): UInt {\n  return random(Random)\n}\n\n/**\n * Returns a random element from this array.\n
* \n * @throws NoSuchElementException if this array is empty.\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.random(): ULong {\n  return random(Random)\n}\n\n/**\n * Returns a random element from this
array.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.random(): UByte {\n  return random(Random)\n}\n\n/**\n * Returns a random element from this
array.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.random(): UShort {\n  return random(Random)\n}\n\n/**\n * Returns a random element from this
array using the specified source of randomness.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.random(random: Random): UInt
{\n  if (isEmpty())\n    throw NoSuchElementException("Array is empty.")\n  return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.random(random: Random):
ULong {\n  if (isEmpty())\n    throw NoSuchElementException("Array is empty.")\n  return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.random(random: Random):
UByte {\n  if (isEmpty())\n    throw NoSuchElementException("Array is empty.")\n  return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness.\n * \n * @throws NoSuchElementException if this array is empty.\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.random(random: Random):
UShort {\n  if (isEmpty())\n    throw NoSuchElementException("Array is empty.")\n  return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array, or `null` if this array is empty.\n
*/\n\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun UIntArray.randomOrNull(): UInt? {\n  return
randomOrNull(Random)\n}\n\n/**\n * Returns a random element from this array, or `null` if this array is empty.\n
*/\n

```

```

*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun ULongArray.randomOrNull(): ULong? {\n    return
randomOrNull(Random)\n}\n\n/**\n * Returns a random element from this array, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun UByteArray.randomOrNull(): UByte? {\n    return
randomOrNull(Random)\n}\n\n/**\n * Returns a random element from this array, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun UShortArray.randomOrNull(): UShort? {\n    return
randomOrNull(Random)\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\np
ublic fun UIntArray.randomOrNull(random: Random): UInt? {\n    if (isEmpty())\n        return null\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\np
ublic fun ULongArray.randomOrNull(random: Random): ULong? {\n    if (isEmpty())\n        return null\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\np
ublic fun UByteArray.randomOrNull(random: Random): UByte? {\n    if (isEmpty())\n        return null\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns a random element from this array using the specified source of
randomness, or `null` if this array is empty.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\np
ublic fun UShortArray.randomOrNull(random: Random): UShort? {\n    if (isEmpty())\n        return null\n    return
get(random.nextInt(size))\n}\n\n/**\n * Returns the single element, or throws an exception if the array is empty or
has more than one element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.single(): UInt {\n    return storage.single().toUInt()\n}\n\n/**\n * Returns the single element, or throws an
exception if the array is empty or has more than one element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.single(): ULong {\n    return storage.single().toULong()\n}\n\n/**\n * Returns the single element, or
throws an exception if the array is empty or has more than one element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.single(): UByte {\n    return storage.single().toUByte()\n}\n\n/**\n * Returns the single element, or
throws an exception if the array is empty or has more than one element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.single(): UShort {\n    return storage.single().toUShort()\n}\n\n/**\n * Returns the single element
matching the given [predicate], or throws exception if there is no or more than one matching element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.single(predicate: (UInt) -> Boolean): UInt {\n    var single: UInt? = null\n    var found = false\n    for
(element in this) {\n        if (predicate(element)) {\n            if (found) throw IllegalArgumentException("Array
contains more than one matching element.")\n            single = element\n            found = true\n        }\n    }\n    if
(!found) throw NoSuchElementException("Array contains no element matching the predicate.")\n}\n
@Suppress("UNCHECKED_CAST")\n    return single as UInt\n}\n\n/**\n * Returns the single element matching
the given [predicate], or throws exception if there is no or more than one matching element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.single(predicate: (ULong) -> Boolean): ULong {\n    var single: ULong? = null\n    var found = false\n
for (element in this) {\n        if (predicate(element)) {\n            if (found) throw IllegalArgumentException("Array

```

```

contains more than one matching element.}")\n        single = element\n        found = true\n    }\n }\n if
(!found) throw NoSuchElementException("Array contains no element matching the predicate.")\n
@Suppress("UNCHECKED_CAST")\n    return single as ULong\n}\n\n/**\n * Returns the single element
matching the given [predicate], or throws exception if there is no or more than one matching element.\n
*\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.single(predicate: (UByte) -> Boolean): UByte {\n    var single: UByte? = null\n    var found = false\n
for (element in this) {\n        if (predicate(element)) {\n            if (found) throw IllegalArgumentException("Array
contains more than one matching element.>")\n                single = element\n                found = true\n            }\n        }\n    if
(!found) throw NoSuchElementException("Array contains no element matching the predicate.")\n
@Suppress("UNCHECKED_CAST")\n    return single as UByte\n}\n\n/**\n * Returns the single element
matching the given [predicate], or throws exception if there is no or more than one matching element.\n
*\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.single(predicate: (UShort) -> Boolean): UShort {\n    var single: UShort? = null\n    var found = false\n
for (element in this) {\n        if (predicate(element)) {\n            if (found) throw IllegalArgumentException("Array
contains more than one matching element.>")\n                single = element\n                found = true\n            }\n        }\n    if
(!found) throw NoSuchElementException("Array contains no element matching the predicate.")\n
@Suppress("UNCHECKED_CAST")\n    return single as UShort\n}\n\n/**\n * Returns single element, or `null` if
the array is empty or has more than one element.\n
*\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.singleOrNull(): UInt? {\n
return if (size == 1) this[0] else null\n}\n\n/**\n * Returns single element, or `null` if the array is empty or has more
than one element.\n
*\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun
ULongArray.singleOrNull(): ULong? {\n    return if (size == 1) this[0] else null\n}\n\n/**\n * Returns single
element, or `null` if the array is empty or has more than one element.\n
*\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.singleOrNull(): UByte? {\n
return if (size == 1) this[0] else null\n}\n\n/**\n * Returns single element, or `null` if the array is empty or has more
than one element.\n
*\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun
UShortArray.singleOrNull(): UShort? {\n    return if (size == 1) this[0] else null\n}\n\n/**\n * Returns the single
element matching the given [predicate], or `null` if element was not found or more than one element was found.\n
*\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.singleOrNull(predicate: (UInt) -> Boolean): UInt? {\n    var single: UInt? = null\n    var found = false\n
for (element in this) {\n        if (predicate(element)) {\n            if (found) return null\n                single = element\n
found = true\n            }\n        }\n    if (!found) return null\n    return single\n}\n\n/**\n * Returns the single element
matching the given [predicate], or `null` if element was not found or more than one element was found.\n
*\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.singleOrNull(predicate: (ULong) -> Boolean): ULong? {\n    var single: ULong? = null\n    var found
= false\n    for (element in this) {\n        if (predicate(element)) {\n            if (found) return null\n                single =
element\n                found = true\n            }\n        }\n    if (!found) return null\n    return single\n}\n\n/**\n * Returns the
single element matching the given [predicate], or `null` if element was not found or more than one element was
found.\n
*\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline
fun UByteArray.singleOrNull(predicate: (UByte) -> Boolean): UByte? {\n    var single: UByte? = null\n    var found
= false\n    for (element in this) {\n        if (predicate(element)) {\n            if (found) return null\n                single =
element\n                found = true\n            }\n        }\n    if (!found) return null\n    return single\n}\n\n/**\n * Returns the
single element matching the given [predicate], or `null` if element was not found or more than one element was
found.\n
*\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline
fun UShortArray.singleOrNull(predicate: (UShort) -> Boolean): UShort? {\n    var single: UShort? = null\n    var
found = false\n    for (element in this) {\n        if (predicate(element)) {\n            if (found) return null\n                single
= element\n                found = true\n            }\n        }\n    if (!found) return null\n    return single\n}\n\n/**\n * Returns a list
containing all elements except first [n] elements.\n
*\n * @throws IllegalArgumentException if [n] is negative.\n
*
```

```

\n * @sample samples.collections.Collections.Transformations.drop\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.drop(n: Int): List<UInt> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n**\n * Returns a list containing all elements except first [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.drop(n: Int): List<ULong> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n**\n * Returns a list containing all elements except first [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.drop(n: Int): List<UByte> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n**\n * Returns a list containing all elements except first [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.drop(n: Int): List<UShort> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return takeLast((size -
n).coerceAtLeast(0))\n}\n\n**\n * Returns a list containing all elements except last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.dropLast(n: Int): List<UInt> {\n
require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return take((size -
n).coerceAtLeast(0))\n}\n\n**\n * Returns a list containing all elements except last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.dropLast(n: Int):
List<ULong> {\n  require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return take((size -
n).coerceAtLeast(0))\n}\n\n**\n * Returns a list containing all elements except last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.dropLast(n: Int): List<UByte>
{\n  require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return take((size -
n).coerceAtLeast(0))\n}\n\n**\n * Returns a list containing all elements except last [n] elements.\n * \n * @throws
IllegalArgumentException if [n] is negative.\n * \n * @sample
samples.collections.Collections.Transformations.drop\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.dropLast(n: Int):
List<UShort> {\n  require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  return take((size -
n).coerceAtLeast(0))\n}\n\n**\n * Returns a list containing all elements except last elements that satisfy the given
[predicate].\n * \n * @sample samples.collections.Collections.Transformations.drop\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.dropLastWhile(predicate: (UInt) -> Boolean): List<UInt> {\n  for (index in lastIndex downTo 0) {\n
if (!predicate(this[index])) {\n      return take(index + 1)\n    }\n  }\n  return emptyList()\n}\n\n**\n * Returns a list containing all elements except last elements that satisfy the given [predicate].\n * \n * @sample
samples.collections.Collections.Transformations.drop\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.dropLastWhile(predicate: (ULong) -> Boolean): List<ULong> {\n  for (index in lastIndex downTo 0)
{\n    if (!predicate(this[index])) {\n      return take(index + 1)\n    }\n  }\n  return emptyList()\n}\n\n**\n

```

\* Returns a list containing all elements except last elements that satisfy the given [predicate].\n \* \n \* @sample samples.collections.Collections.Transformations.drop\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.dropLastWhile(predicate: (UByte) -> Boolean): List<UByte> {\n  for (index in lastIndex downTo 0)
{\n    if (!predicate(this[index])) {\n      return take(index + 1)\n    }\n  }\n  return emptyList()\n}\n\n/**\n
```

\* Returns a list containing all elements except last elements that satisfy the given [predicate].\n \* \n \* @sample samples.collections.Collections.Transformations.drop\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.dropLastWhile(predicate: (UShort) -> Boolean): List<UShort> {\n  for (index in lastIndex downTo
0) {\n    if (!predicate(this[index])) {\n      return take(index + 1)\n    }\n  }\n  return
emptyList()\n}\n\n/**\n
```

\* Returns a list containing all elements except first elements that satisfy the given [predicate].\n \* \n \* @sample samples.collections.Collections.Transformations.drop\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.dropWhile(predicate: (UInt) -> Boolean): List<UInt> {\n  var yielding = false\n  val list =
ArrayList<UInt>()\n  for (item in this)\n    if (yielding)\n      list.add(item)\n    else if (!predicate(item)) {\n
      list.add(item)\n      yielding = true\n    }\n  return list\n}\n\n/**\n
```

\* Returns a list containing all elements except first elements that satisfy the given [predicate].\n \* \n \* @sample samples.collections.Collections.Transformations.drop\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.dropWhile(predicate: (ULong) -> Boolean): List<ULong> {\n  var yielding = false\n  val list =
ArrayList<ULong>()\n  for (item in this)\n    if (yielding)\n      list.add(item)\n    else if (!predicate(item))
{\n      list.add(item)\n      yielding = true\n    }\n  return list\n}\n\n/**\n
```

\* Returns a list containing all elements except first elements that satisfy the given [predicate].\n \* \n \* @sample samples.collections.Collections.Transformations.drop\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.dropWhile(predicate: (UByte) -> Boolean): List<UByte> {\n  var yielding = false\n  val list =
ArrayList<UByte>()\n  for (item in this)\n    if (yielding)\n      list.add(item)\n    else if (!predicate(item))
{\n      list.add(item)\n      yielding = true\n    }\n  return list\n}\n\n/**\n
```

\* Returns a list containing all elements except first elements that satisfy the given [predicate].\n \* \n \* @sample samples.collections.Collections.Transformations.drop\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.dropWhile(predicate: (UShort) -> Boolean): List<UShort> {\n  var yielding = false\n  val list =
ArrayList<UShort>()\n  for (item in this)\n    if (yielding)\n      list.add(item)\n    else if (!predicate(item))
{\n      list.add(item)\n      yielding = true\n    }\n  return list\n}\n\n/**\n
```

\* Returns a list containing only elements matching the given [predicate].\n \* \n \* @sample samples.collections.Collections.Filtering.filter\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.filter(predicate: (UInt) -> Boolean): List<UInt> {\n  return filterTo(ArrayList<UInt>(),
predicate)\n}\n\n/**\n
```

\* Returns a list containing only elements matching the given [predicate].\n \* \n \* @sample samples.collections.Collections.Filtering.filter\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.filter(predicate: (ULong) -> Boolean): List<ULong> {\n  return filterTo(ArrayList<ULong>(),
predicate)\n}\n\n/**\n
```

\* Returns a list containing only elements matching the given [predicate].\n \* \n \* @sample samples.collections.Collections.Filtering.filter\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.filter(predicate: (UByte) -> Boolean): List<UByte> {\n  return filterTo(ArrayList<UByte>(),
predicate)\n}\n\n/**\n
```

\* Returns a list containing only elements matching the given [predicate].\n \* \n \* @sample samples.collections.Collections.Filtering.filter\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
```

```

UShortArray.filter(predicate: (UShort) -> Boolean): List<UShort> {\n  return filterTo(ArrayList<UShort>(),
predicate)\n}\n\n/**\n * Returns a list containing only elements matching the given [predicate].\n * @param
[predicate] function that takes the index of an element and the element itself\n * and returns the result of predicate
evaluation on the element.\n * \n * @sample samples.collections.Collections.Filtering.filterIndexed\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.filterIndexed(predicate: (index: Int, UInt) -> Boolean): List<UInt> {\n  return
filterIndexedTo(ArrayList<UInt>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the
given [predicate].\n * @param [predicate] function that takes the index of an element and the element itself\n * and
returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexed\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.filterIndexed(predicate: (index: Int, ULong) -> Boolean): List<ULong> {\n  return
filterIndexedTo(ArrayList<ULong>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the
given [predicate].\n * @param [predicate] function that takes the index of an element and the element itself\n * and
returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexed\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.filterIndexed(predicate: (index: Int, UByte) -> Boolean): List<UByte> {\n  return
filterIndexedTo(ArrayList<UByte>(), predicate)\n}\n\n/**\n * Returns a list containing only elements matching the
given [predicate].\n * @param [predicate] function that takes the index of an element and the element itself\n * and
returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexed\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.filterIndexed(predicate: (index: Int, UShort) -> Boolean): List<UShort> {\n  return
filterIndexedTo(ArrayList<UShort>(), predicate)\n}\n\n/**\n * Appends all elements matching the given [predicate]
to the given [destination].\n * @param [predicate] function that takes the index of an element and the element
itself\n * and returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexedTo\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <C :
MutableCollection<in UInt>> UIntArray.filterIndexedTo(destination: C, predicate: (index: Int, UInt) -> Boolean): C {\n
forEachIndexed { index, element ->\n    if (predicate(index, element)) destination.add(element)\n  }\n  return destination\n}\n\n/**\n * Appends all elements matching the given [predicate] to the given [destination].\n *
@param [predicate] function that takes the index of an element and the element itself\n * and returns the result of
predicate evaluation on the element.\n * \n * @sample samples.collections.Collections.Filtering.filterIndexedTo\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <C :
MutableCollection<in ULong>> ULongArray.filterIndexedTo(destination: C, predicate: (index: Int, ULong) ->
Boolean): C {\n  forEachIndexed { index, element ->\n    if (predicate(index, element))
destination.add(element)\n  }\n  return destination\n}\n\n/**\n * Appends all elements matching the given
[predicate] to the given [destination].\n * @param [predicate] function that takes the index of an element and the
element itself\n * and returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexedTo\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <C :
MutableCollection<in UByte>> UByteArray.filterIndexedTo(destination: C, predicate: (index: Int, UByte) ->
Boolean): C {\n  forEachIndexed { index, element ->\n    if (predicate(index, element))
destination.add(element)\n  }\n  return destination\n}\n\n/**\n * Appends all elements matching the given
[predicate] to the given [destination].\n * @param [predicate] function that takes the index of an element and the
element itself\n * and returns the result of predicate evaluation on the element.\n * \n * @sample
samples.collections.Collections.Filtering.filterIndexedTo\n

```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <C :
MutableCollection<in UShort>> UShortArray.filterIndexedTo(destination: C, predicate: (index: Int, UShort) ->
Boolean): C {\n  forEachIndexed { index, element ->\n    if (predicate(index, element))
destination.add(element)\n  }\n  return destination\n}\n\n/**\n * Returns a list containing all elements not
matching the given [predicate].\n * \n * @sample samples.collections.Collections.Filtering.filter\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.filterNot(predicate: (UInt) -> Boolean): List<UInt> {\n  return filterNotTo(ArrayList<UInt>(),
predicate)\n}\n\n/**\n * Returns a list containing all elements not matching the given [predicate].\n * \n * @sample
samples.collections.Collections.Filtering.filter\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.filterNot(predicate: (ULong) -> Boolean): List<ULong> {\n  return filterNotTo(ArrayList<ULong>(),
predicate)\n}\n\n/**\n * Returns a list containing all elements not matching the given [predicate].\n * \n * @sample
samples.collections.Collections.Filtering.filter\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.filterNot(predicate: (UByte) -> Boolean): List<UByte> {\n  return filterNotTo(ArrayList<UByte>(),
predicate)\n}\n\n/**\n * Returns a list containing all elements not matching the given [predicate].\n * \n * @sample
samples.collections.Collections.Filtering.filter\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.filterNot(predicate: (UShort) -> Boolean): List<UShort> {\n  return
filterNotTo(ArrayList<UShort>(), predicate)\n}\n\n/**\n * Appends all elements not matching the given [predicate]
to the given [destination].\n * \n * @sample samples.collections.Collections.Filtering.filterTo\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <C :
MutableCollection<in UInt>> UIntArray.filterNotTo(destination: C, predicate: (UInt) -> Boolean): C {\n  for
(element in this) if (!predicate(element)) destination.add(element)\n  return destination\n}\n\n/**\n * Appends all
elements not matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <C :
MutableCollection<in ULong>> ULongArray.filterNotTo(destination: C, predicate: (ULong) -> Boolean): C {\n  for
(element in this) if (!predicate(element)) destination.add(element)\n  return destination\n}\n\n/**\n * Appends
all elements not matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <C :
MutableCollection<in UByte>> UByteArray.filterNotTo(destination: C, predicate: (UByte) -> Boolean): C {\n  for
(element in this) if (!predicate(element)) destination.add(element)\n  return destination\n}\n\n/**\n * Appends all
elements not matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <C :
MutableCollection<in UShort>> UShortArray.filterNotTo(destination: C, predicate: (UShort) -> Boolean): C {\n  for
(element in this) if (!predicate(element)) destination.add(element)\n  return destination\n}\n\n/**\n * Appends
all elements matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <C :
MutableCollection<in UInt>> UIntArray.filterTo(destination: C, predicate: (UInt) -> Boolean): C {\n  for (element
in this) if (predicate(element)) destination.add(element)\n  return destination\n}\n\n/**\n * Appends all elements
matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <C :
MutableCollection<in ULong>> ULongArray.filterTo(destination: C, predicate: (ULong) -> Boolean): C {\n  for

```

```

(element in this) if (predicate(element)) destination.add(element)\n  return destination\n}\n\n/**\n * Appends all
elements matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <C :
MutableCollection<in UByte>> UByteArray.filterTo(destination: C, predicate: (UByte) -> Boolean): C {\n  for
(element in this) if (predicate(element)) destination.add(element)\n  return destination\n}\n\n/**\n * Appends all
elements matching the given [predicate] to the given [destination].\n * \n * @sample
samples.collections.Collections.Filtering.filterTo\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <C :
MutableCollection<in UShort>> UShortArray.filterTo(destination: C, predicate: (UShort) -> Boolean): C {\n  for
(element in this) if (predicate(element)) destination.add(element)\n  return destination\n}\n\n/**\n * Returns a list
containing elements at indices in the specified [indices] range.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.slice(indices: IntRange):
List<UInt> {\n  if (indices.isEmpty()) return listOf()\n  return copyOfRange(indices.start, indices.endInclusive +
1).asList()\n}\n\n/**\n * Returns a list containing elements at indices in the specified [indices] range.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.slice(indices: IntRange):
List<ULong> {\n  if (indices.isEmpty()) return listOf()\n  return copyOfRange(indices.start, indices.endInclusive
+ 1).asList()\n}\n\n/**\n * Returns a list containing elements at indices in the specified [indices] range.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.slice(indices: IntRange):
List<UByte> {\n  if (indices.isEmpty()) return listOf()\n  return copyOfRange(indices.start, indices.endInclusive
+ 1).asList()\n}\n\n/**\n * Returns a list containing elements at indices in the specified [indices] range.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.slice(indices: IntRange):
List<UShort> {\n  if (indices.isEmpty()) return listOf()\n  return copyOfRange(indices.start, indices.endInclusive
+ 1).asList()\n}\n\n/**\n * Returns a list containing elements at specified [indices].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.slice(indices: Iterable<Int>):
List<UInt> {\n  val size = indices.collectionSizeOrDefault(10)\n  if (size == 0) return emptyList()\n  val list =
ArrayList<UInt>(size)\n  for (index in indices) {\n    list.add(get(index))\n  }\n  return list\n}\n\n/**\n *
Returns a list containing elements at specified [indices].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.slice(indices: Iterable<Int>):
List<ULong> {\n  val size = indices.collectionSizeOrDefault(10)\n  if (size == 0) return emptyList()\n  val list =
ArrayList<ULong>(size)\n  for (index in indices) {\n    list.add(get(index))\n  }\n  return list\n}\n\n/**\n *
Returns a list containing elements at specified [indices].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.slice(indices: Iterable<Int>):
List<UByte> {\n  val size = indices.collectionSizeOrDefault(10)\n  if (size == 0) return emptyList()\n  val list =
ArrayList<UByte>(size)\n  for (index in indices) {\n    list.add(get(index))\n  }\n  return list\n}\n\n/**\n *
Returns a list containing elements at specified [indices].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.slice(indices: Iterable<Int>):
List<UShort> {\n  val size = indices.collectionSizeOrDefault(10)\n  if (size == 0) return emptyList()\n  val list =
ArrayList<UShort>(size)\n  for (index in indices) {\n    list.add(get(index))\n  }\n  return list\n}\n\n/**\n *
Returns an array containing elements of this array at specified [indices].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.sliceArray(indices:
Collection<Int>): UIntArray {\n  return UIntArray(storage.sliceArray(indices))\n}\n\n/**\n * Returns an array
containing elements of this array at specified [indices].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.sliceArray(indices:
Collection<Int>): ULongArray {\n  return ULongArray(storage.sliceArray(indices))\n}\n\n/**\n * Returns an array
containing elements of this array at specified [indices].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.sliceArray(indices:
Collection<Int>): UByteArray {\n  return UByteArray(storage.sliceArray(indices))\n}\n\n/**\n * Returns an array
containing elements of this array at specified [indices].\n

```

containing elements of this array at specified [indices].\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.sliceArray(indices:  
Collection<Int>): UShortArray {\n    return UShortArray(storage.sliceArray(indices))\n}\n\n/**\n * Returns an  
array containing elements at indices in the specified [indices] range.\n
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.sliceArray(indices: IntRange):  
UIntArray {\n    return UIntArray(storage.sliceArray(indices))\n}\n\n/**\n * Returns an array containing elements at  
indices in the specified [indices] range.\n
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun  
ULongArray.sliceArray(indices: IntRange): ULongArray {\n    return  
ULongArray(storage.sliceArray(indices))\n}\n\n/**\n * Returns an array containing elements at indices in the  
specified [indices] range.\n
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun  
UByteArray.sliceArray(indices: IntRange): UByteArray {\n    return  
UByteArray(storage.sliceArray(indices))\n}\n\n/**\n * Returns an array containing elements at indices in the  
specified [indices] range.\n
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun  
UShortArray.sliceArray(indices: IntRange): UShortArray {\n    return  
UShortArray(storage.sliceArray(indices))\n}\n\n/**\n * Returns a list containing first [n] elements.\n * \n *  
@throws IllegalArgumentException if [n] is negative.\n * \n * @sample
```

```
samples.collections.Collections.Transformations.take\n
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.take(n: Int): List<UInt> {\n    require(n >= 0) { "Requested element count $n is less than zero." }\n    if (n == 0) return emptyList()\n    if (n >=  
size) return toList()\n    if (n == 1) return listOf(this[0])\n    var count = 0\n    val list = ArrayList<UInt>(n)\n    for  
(item in this) {\n        list.add(item)\n        if (++count == n)\n            break\n    }\n    return list\n}\n\n/**\n * Returns  
a list containing first [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
```

```
samples.collections.Collections.Transformations.take\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.take(n: Int): List<ULong> {\n    require(n >= 0) { "Requested element count $n is less than zero." }\n    if (n == 0) return emptyList()\n    if (n >=  
size) return toList()\n    if (n == 1) return listOf(this[0])\n    var count = 0\n    val list = ArrayList<ULong>(n)\n    for  
(item in this) {\n        list.add(item)\n        if (++count == n)\n            break\n    }\n    return list\n}\n\n/**\n * Returns  
a list containing first [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
```

```
samples.collections.Collections.Transformations.take\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.take(n: Int): List<UShort> {\n    require(n >= 0) { "Requested element count $n is less than zero." }\n    if (n == 0) return emptyList()\n    if (n >=  
size) return toList()\n    if (n == 1) return listOf(this[0])\n    var count = 0\n    val list = ArrayList<UShort>(n)\n    for  
(item in this) {\n        list.add(item)\n        if (++count == n)\n            break\n    }\n    return list\n}\n\n/**\n * Returns  
a list containing first [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
```

```
samples.collections.Collections.Transformations.take\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.take(n: Int): List<UShort> {\n    require(n >= 0) { "Requested element count $n is less than zero." }\n    if (n == 0) return emptyList()\n    if (n >=  
size) return toList()\n    if (n == 1) return listOf(this[0])\n    var count = 0\n    val list = ArrayList<UShort>(n)\n    for  
(item in this) {\n        list.add(item)\n        if (++count == n)\n            break\n    }\n    return list\n}\n\n/**\n * Returns  
a list containing last [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
```

```
samples.collections.Collections.Transformations.take\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.takeLast(n: Int): List<UInt> {\n    require(n >= 0) { "Requested element count $n is less than zero." }\n    if (n == 0) return emptyList()\n    val size =  
size\n    if (n >= size) return toList()\n    if (n == 1) return listOf(this[size - 1])\n    val list = ArrayList<UInt>(n)\n    for (index in size - n until size)\n        list.add(this[index])\n    return list\n}\n\n/**\n * Returns a list containing last  
[n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample
```

```
samples.collections.Collections.Transformations.take\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.takeLast(n: Int): List<ULong>
```

```

{\n  require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  if (n == 0) return emptyList()\n  val size = size\n  if (n >= size) return toList()\n  if (n == 1) return listOf(this[size - 1])\n  val list = ArrayList<ULong>(n)\n  for (index in size - n until size)\n    list.add(this[index])\n  return list\n}\n\n/**\n * Returns a list containing last [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample samples.collections.Collections.Transformations.take\n */\n\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.takeLast(n: Int): List<UByte> {\n  require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  if (n == 0) return emptyList()\n  val size = size\n  if (n >= size) return toList()\n  if (n == 1) return listOf(this[size - 1])\n  val list = ArrayList<UByte>(n)\n  for (index in size - n until size)\n    list.add(this[index])\n  return list\n}\n\n/**\n * Returns a list containing last [n] elements.\n * \n * @throws IllegalArgumentException if [n] is negative.\n * \n * @sample samples.collections.Collections.Transformations.take\n */\n\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.takeLast(n: Int): List<UShort> {\n  require(n >= 0) { \"Requested element count $n is less than zero.\" }\n  if (n == 0) return emptyList()\n  val size = size\n  if (n >= size) return toList()\n  if (n == 1) return listOf(this[size - 1])\n  val list = ArrayList<UShort>(n)\n  for (index in size - n until size)\n    list.add(this[index])\n  return list\n}\n\n/**\n * Returns a list containing last elements satisfying the given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n */\n\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UIntArray.takeLastWhile(predicate: (UInt) -> Boolean): List<UInt> {\n  for (index in lastIndex downTo 0) {\n    if (!predicate(this[index]))\n      return drop(index + 1)\n  }\n  return toList()\n}\n\n/**\n * Returns a list containing last elements satisfying the given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n */\n\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun ULongArray.takeLastWhile(predicate: (ULong) -> Boolean): List<ULong> {\n  for (index in lastIndex downTo 0) {\n    if (!predicate(this[index]))\n      return drop(index + 1)\n  }\n  return toList()\n}\n\n/**\n * Returns a list containing last elements satisfying the given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n */\n\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UByteArray.takeLastWhile(predicate: (UByte) -> Boolean): List<UByte> {\n  for (index in lastIndex downTo 0) {\n    if (!predicate(this[index]))\n      return drop(index + 1)\n  }\n  return toList()\n}\n\n/**\n * Returns a list containing last elements satisfying the given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n */\n\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UShortArray.takeLastWhile(predicate: (UShort) -> Boolean): List<UShort> {\n  for (index in lastIndex downTo 0) {\n    if (!predicate(this[index]))\n      return drop(index + 1)\n  }\n  return toList()\n}\n\n/**\n * Returns a list containing first elements satisfying the given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n */\n\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UIntArray.takeWhile(predicate: (UInt) -> Boolean): List<UInt> {\n  val list = ArrayList<UInt>()\n  for (item in this) {\n    if (!predicate(item))\n      break\n    list.add(item)\n  }\n  return list\n}\n\n/**\n * Returns a list containing first elements satisfying the given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n */\n\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun ULongArray.takeWhile(predicate: (ULong) -> Boolean): List<ULong> {\n  val list = ArrayList<ULong>()\n  for (item in this) {\n    if (!predicate(item))\n      break\n    list.add(item)\n  }\n  return list\n}\n\n/**\n * Returns a list containing first elements satisfying the given [predicate].\n * \n * @sample samples.collections.Collections.Transformations.take\n */\n\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun

```

```

ByteArray.takeWhile(predicate: (UByte) -> Boolean): List<UByte> {\n  val list = ArrayList<UByte>()\n  for\n  (item in this) {\n    if (!predicate(item))\n      break\n    list.add(item)\n  }\n  return list\n}\n\nReturns a list containing first elements satisfying the given [predicate].\n\n * \n * @sample\n samples.collections.Collections.Transformations.take\n\n *\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nUShortArray.takeWhile(predicate: (UShort) -> Boolean): List<UShort> {\n  val list = ArrayList<UShort>()\n  for\n  (item in this) {\n    if (!predicate(item))\n      break\n    list.add(item)\n  }\n  return list\n}\n\nReverses elements in the array in-place.\n\n *\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nUIntArray.reverse(): Unit {\n  storage.reverse()\n}\n\nReverses elements in the array in-place.\n\n *\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nULongArray.reverse(): Unit {\n  storage.reverse()\n}\n\nReverses elements in the array in-place.\n\n *\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nUByteArray.reverse(): Unit {\n  storage.reverse()\n}\n\nReverses elements in the array in-place.\n\n *\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nUShortArray.reverse(fromIndex: Int, toIndex: Int): Unit {\n  storage.reverse(fromIndex, toIndex)\n}\n\nReverses elements of the array in the specified\nrange in-place.\n\n * \n * @param fromIndex the start of the range (inclusive) to reverse.\n * @param toIndex the end\nof the range (exclusive) to reverse.\n\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero\nor [toIndex] is greater than the size of this array.\n\n * \n * @throws IllegalArgumentException if [fromIndex] is greater\nthan [toIndex].\n\n *\n\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic\ninline fun UIntArray.reverse(fromIndex: Int, toIndex: Int): Unit {\n  storage.reverse(fromIndex,\ntoIndex)\n}\n\nReverses elements of the array in the specified range in-place.\n\n * \n * @param fromIndex\nthe start of the range (inclusive) to reverse.\n\n * @param toIndex the end of the range (exclusive) to reverse.\n\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this\narray.\n\n * \n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n\n *\n\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nULongArray.reverse(fromIndex: Int, toIndex: Int): Unit {\n  storage.reverse(fromIndex, toIndex)\n}\n\nReverses elements\nof the array in the specified range in-place.\n\n * \n * @param fromIndex the start of the range\n(inclusive) to reverse.\n\n * @param toIndex the end of the range (exclusive) to reverse.\n\n * \n * @throws\nIndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n\n * \n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n\n *\n\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nUByteArray.reverse(fromIndex: Int, toIndex: Int): Unit {\n  storage.reverse(fromIndex, toIndex)\n}\n\nReverses elements of the array in the specified range in-place.\n\n * \n * @param fromIndex the start of the range\n(inclusive) to reverse.\n\n * @param toIndex the end of the range (exclusive) to reverse.\n\n * \n * @throws\nIndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n\n * \n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n\n *\n\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nUShortArray.reverse(fromIndex: Int, toIndex: Int): Unit {\n  storage.reverse(fromIndex, toIndex)\n}\n\nReturns a list with elements in reversed order.\n\n *\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic\nfun UIntArray.reversed(): List<UInt> {\n  if (isEmpty()) return emptyList()\n  val list = toMutableList()\n  list.reverse()\n  return list\n}\n\nReturns a list with elements in reversed order.\n\n *\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun\nULongArray.reversed(): List<ULong> {\n  if (isEmpty()) return emptyList()\n  val list = toMutableList()\n  list.reverse()\n  return list\n}\n\nReturns\na list with elements in reversed order.\n\n *\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun\nUByteArray.reversed(): List<UByte> {\n  if (isEmpty()) return emptyList()\n  val list = toMutableList()\n  list.reverse()\n  return list\n}\n\nReturns a list with elements in reversed order.\n\n *\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun\nUShortArray.reversed(): List<UShort> {\n

```

```

if (isEmpty()) return emptyList()\n    val list = toMutableList()\n    list.reverse()\n    return list\n}\n\n/**\n * Returns
an array with elements of this array in reversed order.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.reversedArray(): UIntArray {\n    return UIntArray(storage.reversedArray())\n}\n\n/**\n * Returns an
array with elements of this array in reversed order.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.reversedArray(): ULongArray {\n    return ULongArray(storage.reversedArray())\n}\n\n/**\n *
Returns an array with elements of this array in reversed order.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.reversedArray(): UByteArray {\n    return UByteArray(storage.reversedArray())\n}\n\n/**\n * Returns
an array with elements of this array in reversed order.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.reversedArray(): UShortArray {\n    return UShortArray(storage.reversedArray())\n}\n\n/**\n *
Randomly shuffles elements in this array in-place.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.shuffle(): Unit {\n
shuffle(Random)\n}\n\n/**\n * Randomly shuffles elements in this array in-place.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.shuffle(): Unit {\n
shuffle(Random)\n}\n\n/**\n * Randomly shuffles elements in this array in-place.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.shuffle(): Unit {\n
shuffle(Random)\n}\n\n/**\n * Randomly shuffles elements in this array in-place.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.shuffle(): Unit {\n
shuffle(Random)\n}\n\n/**\n * Randomly shuffles elements in this array in-place using the specified [random]
instance as the source of randomness.\n * \n * See:
https://en.wikipedia.org/wiki/Fisher%20%80%93Yates\_shuffle#The\_modern\_algorithm\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.shuffle(random: Random): Unit
{\n    for (i in lastIndex downTo 1) {\n        val j = random.nextInt(i + 1)\n        val copy = this[i]\n        this[i] =
this[j]\n        this[j] = copy\n    }\n}\n\n/**\n * Randomly shuffles elements in this array in-place using the specified
[random] instance as the source of randomness.\n * \n * See:
https://en.wikipedia.org/wiki/Fisher%20%80%93Yates\_shuffle#The\_modern\_algorithm\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.shuffle(random: Random):
Unit {\n    for (i in lastIndex downTo 1) {\n        val j = random.nextInt(i + 1)\n        val copy = this[i]\n        this[i] =
this[j]\n        this[j] = copy\n    }\n}\n\n/**\n * Randomly shuffles elements in this array in-place using the specified
[random] instance as the source of randomness.\n * \n * See:
https://en.wikipedia.org/wiki/Fisher%20%80%93Yates\_shuffle#The\_modern\_algorithm\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.shuffle(random: Random):
Unit {\n    for (i in lastIndex downTo 1) {\n        val j = random.nextInt(i + 1)\n        val copy = this[i]\n        this[i] =
this[j]\n        this[j] = copy\n    }\n}\n\n/**\n * Randomly shuffles elements in this array in-place using the specified
[random] instance as the source of randomness.\n * \n * See:
https://en.wikipedia.org/wiki/Fisher%20%80%93Yates\_shuffle#The\_modern\_algorithm\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.shuffle(random: Random):
Unit {\n    for (i in lastIndex downTo 1) {\n        val j = random.nextInt(i + 1)\n        val copy = this[i]\n        this[i] =
this[j]\n        this[j] = copy\n    }\n}\n\n/**\n * Sorts elements in the array in-place descending according to their
natural sort order.\n *\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun
UIntArray.sortDescending(): Unit {\n    if (size > 1) {\n        sort()\n        reverse()\n    }\n}\n\n/**\n * Sorts elements
in the array in-place descending according to their natural sort order.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun
ULongArray.sortDescending(): Unit {\n    if (size > 1) {\n        sort()\n        reverse()\n    }\n}\n\n/**\n * Sorts elements
in the array in-place descending according to their natural sort order.\n *\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun

```

```

UByteArray.sortDescending(): Unit {n  if (size > 1) {n      sort()\n      reverse()\n  }n}\n\n/**\n * Sorts elements in the array in-place descending according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.sortDescending(): Unit {n  if (size > 1) {n      sort()\n      reverse()\n  }n}\n\n/**\n * Returns a list of all elements sorted according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.sorted(): List<UInt> {n  return copyOf().apply { sort() }.asList()\n}\n\n/**\n * Returns a list of all elements sorted according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.sorted(): List<ULong> {n  return copyOf().apply { sort() }.asList()\n}\n\n/**\n * Returns a list of all elements sorted according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.sorted(): List<UByte> {n  return copyOf().apply { sort() }.asList()\n}\n\n/**\n * Returns a list of all elements sorted according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.sorted(): List<UShort> {n  return copyOf().apply { sort() }.asList()\n}\n\n/**\n * Returns an array with all elements of this array sorted according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.sortedArray(): UIntArray {n  if (isEmpty()) return this\n  return this.copyOf().apply { sort() }\n}\n\n/**\n * Returns an array with all elements of this array sorted according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.sortedArray(): ULongArray {n  if (isEmpty()) return this\n  return this.copyOf().apply { sort() }\n}\n\n/**\n * Returns an array with all elements of this array sorted according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.sortedArray(): UByteArray {n  if (isEmpty()) return this\n  return this.copyOf().apply { sort() }\n}\n\n/**\n * Returns an array with all elements of this array sorted according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.sortedArray(): UShortArray {n  if (isEmpty()) return this\n  return this.copyOf().apply { sort() }\n}\n\n/**\n * Returns an array with all elements of this array sorted descending according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.sortedArrayDescending(): UIntArray {n  if (isEmpty()) return this\n  return this.copyOf().apply { sortDescending() }\n}\n\n/**\n * Returns an array with all elements of this array sorted descending according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.sortedArrayDescending(): ULongArray {n  if (isEmpty()) return this\n  return this.copyOf().apply { sortDescending() }\n}\n\n/**\n * Returns an array with all elements of this array sorted descending according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.sortedArrayDescending(): UByteArray {n  if (isEmpty()) return this\n  return this.copyOf().apply { sortDescending() }\n}\n\n/**\n * Returns an array with all elements of this array sorted descending according to their natural sort order.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.sortedArrayDescending(): UShortArray {n  if (isEmpty()) return this\n  return this.copyOf().apply { sortDescending() }\n}\n\n/**\n * Returns a list of all elements sorted descending according to their natural sort order.\n\n*\n * The sort is stable. It means that equal elements preserve their order relative to each other after sorting.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.sortedDescending(): List<UInt> {n  return copyOf().apply { sort() }.reversed()\n}\n\n/**\n * Returns a list of all elements sorted descending according to their natural sort order.\n\n*\n * The sort is stable. It means that equal elements preserve their order relative to each other after sorting.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.sortedDescending(): List<ULong> {n  return copyOf().apply { sort() }.reversed()\n}\n\n/**\n * Returns a list of all elements sorted descending according to their natural sort order.\n\n*\n * The sort is stable. It means that equal elements preserve their order relative to each other after sorting.\n\n*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.sortedDescending(): List<UByte> {n  return copyOf().apply { sort() }.reversed()\n}\n\n/**\n * Returns a list of all elements sorted

```

descending according to their natural sort order.\n \* \n \* The sort is `_stable_`. It means that equal elements preserve their order relative to each other after sorting.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.sortedDescending(): List<UShort> {\n return copyOf().apply { sort() }.reversed()\n}\n\n/\*\*\n \* Returns an array of type [ByteArray], which is a view of this array where each element is a signed reinterpretation\n \* of the corresponding element of this array.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UByteArray.asByteArray(): ByteArray {\n return storage\n}\n\n/\*\*\n \* Returns an array of type [IntArray], which is a view of this array where each element is a signed reinterpretation\n \* of the corresponding element of this array.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UIntArray.asIntArray(): IntArray {\n return storage\n}\n\n/\*\*\n \* Returns a [List] that wraps the original array.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic expect fun UIntArray.asList(): List<UInt>\n\n/\*\*\n \* Returns a [List] that wraps the original array.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic expect fun ULongArray.asList(): List<ULong>\n\n/\*\*\n \* Returns a [List] that wraps the original array.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic expect fun UByteArray.asList(): List<UByte>\n\n/\*\*\n \* Returns a [List] that wraps the original array.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic expect fun UShortArray.asList(): List<UShort>\n\n/\*\*\n \* Returns an array of type [LongArray], which is a view of this array where each element is a signed reinterpretation\n \* of the corresponding element of this array.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun ULongArray.asLongArray(): LongArray {\n return storage\n}\n\n/\*\*\n \* Returns an array of type [ShortArray], which is a view of this array where each element is a signed reinterpretation\n \* of the corresponding element of this array.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UShortArray.asShortArray(): ShortArray {\n return storage\n}\n\n/\*\*\n \* Returns an array of type [UByteArray], which is a view of this array where each element is an unsigned reinterpretation\n \* of the corresponding element of this array.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun ByteArray.asUByteArray(): UByteArray {\n return UByteArray(this)\n}\n\n/\*\*\n \* Returns an array of type [UIntArray], which is a view of this array where each element is an unsigned reinterpretation\n \* of the corresponding element of this array.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun IntArray.asUIntArray(): UIntArray {\n return UIntArray(this)\n}\n\n/\*\*\n \* Returns an array of type [ULongArray], which is a view of this array where each element is an unsigned reinterpretation\n \* of the corresponding element of this array.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun LongArray.asULongArray(): ULongArray {\n return ULongArray(this)\n}\n\n/\*\*\n \* Returns an array of type [UShortArray], which is a view of this array where each element is an unsigned reinterpretation\n \* of the corresponding element of this array.\n \*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun ShortArray.asUShortArray(): UShortArray {\n return UShortArray(this)\n}\n\n/\*\*\n \* Returns `true` if the two specified arrays are *structurally* equal to one another,\n \* i.e. contain the same number of the same elements in the same order.\n \*/\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")\n@SinceKotlin("1.3")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\n@ExperimentalUnsignedTypes\npublic infix fun UIntArray.contentEquals(other: UIntArray): Boolean {\n return this.contentEquals(other)\n}\n\n/\*\*\n \* Returns `true` if the two specified arrays are *structurally* equal to one another,\n \* i.e. contain the same number of the same elements in the same order.\n \*/\n@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation warning.")\n@SinceKotlin("1.3")\n@DeprecatedSinceKotlin(hiddenSince = "1.4")\n@ExperimentalUnsignedTypes\npublic infix fun ULongArray.contentEquals(other: ULongArray):

```

Boolean {
    return this.contentEquals(other)
}

/**
 * Returns `true` if the two specified arrays are
 *structurally* equal to one another,
 * i.e. contain the same number of the same elements in the same order.
 */
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.")
@SinceKotlin("1.3")
@DeprecatedSinceKotlin(hiddenSince =
"1.4")
@ExperimentalUnsignedTypes
public infix fun UByteArray.contentEquals(other: UByteArray): Boolean
{
    return this.contentEquals(other)
}

/**
 * Returns `true` if the two specified arrays are
 *structurally* equal
 to one another,
 * i.e. contain the same number of the same elements in the same order.
 */
@Deprecated("Use
Kotlin compiler 1.4 to avoid deprecation
warning.")
@SinceKotlin("1.3")
@DeprecatedSinceKotlin(hiddenSince =
"1.4")
@ExperimentalUnsignedTypes
public infix fun UShortArray.contentEquals(other: UShortArray):
Boolean
{
    return this.contentEquals(other)
}

/**
 * Returns `true` if the two specified arrays are
 *structurally* equal
 to one another,
 * i.e. contain the same number of the same elements in the same order.
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
public infix fun UIntArray?.contentEquals(other:
UIntArray?): Boolean
{
    return this?.storage?.contentEquals(other?.storage)
}

/**
 * Returns `true` if the two
 specified arrays are
 *structurally* equal to one another,
 * i.e. contain the same number of the same elements in the
 same order.
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
public infix fun
ULongArray?.contentEquals(other: ULongArray?): Boolean
{
    return
this?.storage?.contentEquals(other?.storage)
}

/**
 * Returns `true` if the two specified arrays are
 *structurally*
 equal to one another,
 * i.e. contain the same number of the same elements in the same order.
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
public infix fun UByteArray?.contentEquals(other:
UByteArray?): Boolean
{
    return this?.storage?.contentEquals(other?.storage)
}

/**
 * Returns `true` if the
 two specified arrays are
 *structurally* equal to one another,
 * i.e. contain the same number of the same elements
 in the same order.
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
public infix fun
UShortArray?.contentEquals(other: UShortArray?): Boolean
{
    return
this?.storage?.contentEquals(other?.storage)
}

/**
 * Returns a hash code based on the contents of this array as
 if it is [List].
 */
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.")
@SinceKotlin("1.3")
@DeprecatedSinceKotlin(hiddenSince =
"1.4")
@ExperimentalUnsignedTypes
public fun UIntArray.contentHashCode(): Int
{
    return
this.contentHashCode()
}

/**
 * Returns a hash code based on the contents of this array as if it is [List].
 */
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.")
@SinceKotlin("1.3")
@DeprecatedSinceKotlin(hiddenSince =
"1.4")
@ExperimentalUnsignedTypes
public fun ULongArray.contentHashCode(): Int
{
    return
this.contentHashCode()
}

/**
 * Returns a hash code based on the contents of this array as if it is [List].
 */
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.")
@SinceKotlin("1.3")
@DeprecatedSinceKotlin(hiddenSince =
"1.4")
@ExperimentalUnsignedTypes
public fun UByteArray.contentHashCode(): Int
{
    return
this.contentHashCode()
}

/**
 * Returns a hash code based on the contents of this array as if it is [List].
 */
@Deprecated("Use Kotlin compiler 1.4 to avoid deprecation
warning.")
@SinceKotlin("1.3")
@DeprecatedSinceKotlin(hiddenSince =
"1.4")
@ExperimentalUnsignedTypes
public fun UShortArray.contentHashCode(): Int
{
    return
this.contentHashCode()
}

/**
 * Returns a hash code based on the contents of this array as if it is [List].
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
public fun UIntArray?.contentHashCode(): Int
{
    return this?.storage?.contentHashCode()
}

/**
 * Returns a hash code based on the contents of this array as if it
 is [List].
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
public fun
ULongArray?.contentHashCode(): Int
{
    return this?.storage?.contentHashCode()
}

/**
 * Returns a hash
 code based on the contents of this array as if it is [List].
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
public fun UByteArray?.contentHashCode(): Int
{
    return this?.storage?.contentHashCode()
}

/**
 * Returns a hash code based on the contents of this array as if it

```

```

is [List].\n * \n @SinceKotlin("1.4")\n @ExperimentalUnsignedTypes\n public fun
UShortArray?.contentHashCode(): Int {\n    return this?.storage.contentHashCode()\n}\n\n/**\n * Returns a string
representation of the contents of the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n * \n @Deprecated("Use Kotlin compiler 1.4 to
avoid deprecation warning.")\n @SinceKotlin("1.3")\n @DeprecatedSinceKotlin(hiddenSince =
"1.4")\n @ExperimentalUnsignedTypes\n public fun UIntArray.contentToString(): String {\n    return
this.contentToString()\n}\n\n/**\n * Returns a string representation of the contents of the specified array as if it is
[List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n * \n @Deprecated("Use
Kotlin compiler 1.4 to avoid deprecation
warning.")\n @SinceKotlin("1.3")\n @DeprecatedSinceKotlin(hiddenSince =
"1.4")\n @ExperimentalUnsignedTypes\n public fun ULongArray.contentToString(): String {\n    return
this.contentToString()\n}\n\n/**\n * Returns a string representation of the contents of the specified array as if it is
[List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n * \n @Deprecated("Use
Kotlin compiler 1.4 to avoid deprecation
warning.")\n @SinceKotlin("1.3")\n @DeprecatedSinceKotlin(hiddenSince =
"1.4")\n @ExperimentalUnsignedTypes\n public fun UByteArray.contentToString(): String {\n    return
this.contentToString()\n}\n\n/**\n * Returns a string representation of the contents of the specified array as if it is
[List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n * \n @Deprecated("Use
Kotlin compiler 1.4 to avoid deprecation
warning.")\n @SinceKotlin("1.3")\n @DeprecatedSinceKotlin(hiddenSince =
"1.4")\n @ExperimentalUnsignedTypes\n public fun UShortArray.contentToString(): String {\n    return
this.contentToString()\n}\n\n/**\n * Returns a string representation of the contents of the specified array as if it is
[List].\n * \n * @sample samples.collections.Arrays.ContentOperations.contentToString\n * \n @Deprecated("Use
Kotlin compiler 1.4 to avoid deprecation
warning.")\n @SinceKotlin("1.4")\n @ExperimentalUnsignedTypes\n public fun UIntArray?.contentToString(): String {\n
return this?.joinToString(", ", "[", "]") ?: "null"\n}\n\n/**\n * Returns a string representation of the contents of
the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n * \n @SinceKotlin("1.4")\n @ExperimentalUnsignedTypes\n public fun ULongArray?.contentToString(): String {\n
return this?.joinToString(", ", "[", "]") ?: "null"\n}\n\n/**\n * Returns a string representation of the contents of
the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n * \n @SinceKotlin("1.4")\n @ExperimentalUnsignedTypes\n public fun UByteArray?.contentToString(): String {\n
return this?.joinToString(", ", "[", "]") ?: "null"\n}\n\n/**\n * Returns a string representation of the contents of
the specified array as if it is [List].\n * \n * @sample
samples.collections.Arrays.ContentOperations.contentToString\n * \n @SinceKotlin("1.4")\n @ExperimentalUnsignedTypes\n public fun UShortArray?.contentToString(): String {\n
return this?.joinToString(", ", "[", "]") ?: "null"\n}\n\n/**\n * Copies this array or its subrange into the
[destination] array and returns that array.\n * \n * It's allowed to pass the same array in the [destination] and even
specify the subrange so that it overlaps with the destination range.\n * \n * @param destination the array to copy
to.\n * @param destinationOffset the position in the [destination] array to copy to, 0 by default.\n * @param
startIndex the beginning (inclusive) of the subrange to copy, 0 by default.\n * @param endIndex the end (exclusive)
of the subrange to copy, size of this array by default.\n * @throws IndexOutOfBoundsException or
[IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex
> endIndex`.\n * @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array
starting at the specified [destinationOffset],\n * or when that index is out of the [destination] array indices range.\n *
\n * @return the [destination] array.\n
\n * \n @SinceKotlin("1.3")\n @ExperimentalUnsignedTypes\n @kotlin.internal.InlineOnly\n public inline fun
UIntArray.copyInto(destination: UIntArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size):

```

```

UIntArray {
    storage.copyInto(destination.storage, destinationOffset, startIndex, endIndex)
} return destination

/**
 * Copies this array or its subrange into the [destination] array and returns that array.
 * It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range.
 * @param destination the array to copy to.
 * @param destinationOffset the position in the [destination] array to copy to, 0 by default.
 * @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default.
 * @param endIndex the end (exclusive) of the subrange to copy, size of this array by default.
 * @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`.
 * @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],
 * or when that index is out of the [destination] array indices range.
 * @return the [destination] array.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
ULongArray.copyInto(destination: ULongArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size):
    ULongArray {
    storage.copyInto(destination.storage, destinationOffset, startIndex, endIndex)
} return destination

/**
 * Copies this array or its subrange into the [destination] array and returns that array.
 * It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range.
 * @param destination the array to copy to.
 * @param destinationOffset the position in the [destination] array to copy to, 0 by default.
 * @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default.
 * @param endIndex the end (exclusive) of the subrange to copy, size of this array by default.
 * @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`.
 * @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],
 * or when that index is out of the [destination] array indices range.
 * @return the [destination] array.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UByteArray.copyInto(destination: UByteArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size):
    UByteArray {
    storage.copyInto(destination.storage, destinationOffset, startIndex, endIndex)
} return destination

/**
 * Copies this array or its subrange into the [destination] array and returns that array.
 * It's allowed to pass the same array in the [destination] and even specify the subrange so that it overlaps with the destination range.
 * @param destination the array to copy to.
 * @param destinationOffset the position in the [destination] array to copy to, 0 by default.
 * @param startIndex the beginning (inclusive) of the subrange to copy, 0 by default.
 * @param endIndex the end (exclusive) of the subrange to copy, size of this array by default.
 * @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this array indices or when `startIndex > endIndex`.
 * @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],
 * or when that index is out of the [destination] array indices range.
 * @return the [destination] array.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UShortArray.copyInto(destination: UShortArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = size): UShortArray {
    storage.copyInto(destination.storage, destinationOffset, startIndex, endIndex)
} return destination

/**
 * Returns new array which is a copy of the original array.
 * @sample samples.collections.Arrays.CopyOfOperations.copyOf
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UIntArray.copyOf(): UIntArray {
    return UIntArray(storage.copyOf())
}

/**
 * Returns new array which is a copy of the original array.
 * @sample samples.collections.Arrays.CopyOfOperations.copyOf
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
ULongArray.copyOf(): ULongArray {
    return ULongArray(storage.copyOf())
}

/**
 * Returns new array which is a copy of the original array.
 * @sample samples.collections.Arrays.CopyOfOperations.copyOf
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UByteArray.copyOf(): UByteArray {
    return UByteArray(storage.copyOf())
}

```

\*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UShortArray.copyOf(): UShortArray {\n return UShortArray(storage.copyOf())\n}\n\n/\*\*\n \* Returns new array which is a copy of the original array, resized to the given [newSize].\n \* The copy is either truncated or padded at the end with zero values if necessary.\n \* - If [newSize] is less than the size of the original array, the copy array is truncated to the [newSize].\n \* - If [newSize] is greater than the size of the original array, the extra elements in the copy array are filled with zero values.\n

\*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UIntArray.copyOf(newSize: Int): UIntArray {\n return UIntArray(storage.copyOf(newSize))\n}\n\n/\*\*\n \* Returns new array which is a copy of the original array, resized to the given [newSize].\n \* The copy is either truncated or padded at the end with zero values if necessary.\n \* - If [newSize] is less than the size of the original array, the copy array is truncated to the [newSize].\n \* - If [newSize] is greater than the size of the original array, the extra elements in the copy array are filled with zero values.\n

\*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun ULongArray.copyOf(newSize: Int): ULongArray {\n return ULongArray(storage.copyOf(newSize))\n}\n\n/\*\*\n \* Returns new array which is a copy of the original array, resized to the given [newSize].\n \* The copy is either truncated or padded at the end with zero values if necessary.\n \* - If [newSize] is less than the size of the original array, the copy array is truncated to the [newSize].\n \* - If [newSize] is greater than the size of the original array, the extra elements in the copy array are filled with zero values.\n

\*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UByteArray.copyOf(newSize: Int): UByteArray {\n return UByteArray(storage.copyOf(newSize))\n}\n\n/\*\*\n \* Returns new array which is a copy of the original array, resized to the given [newSize].\n \* The copy is either truncated or padded at the end with zero values if necessary.\n \* - If [newSize] is less than the size of the original array, the copy array is truncated to the [newSize].\n \* - If [newSize] is greater than the size of the original array, the extra elements in the copy array are filled with zero values.\n

\*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UShortArray.copyOfRange(fromIndex: Int, toIndex: Int): UShortArray {\n return UShortArray(storage.copyOfRange(fromIndex, toIndex))\n}\n\n/\*\*\n \* Returns a new array which is a copy of the specified range of the original array.\n \* @param fromIndex the start of the range (inclusive) to copy.\n \* @param toIndex the end of the range (exclusive) to copy.\n \* @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n

\*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UIntArray.copyOfRange(fromIndex: Int, toIndex: Int): UIntArray {\n return UIntArray(storage.copyOfRange(fromIndex, toIndex))\n}\n\n/\*\*\n \* Returns a new array which is a copy of the specified range of the original array.\n \* @param fromIndex the start of the range (inclusive) to copy.\n \* @param toIndex the end of the range (exclusive) to copy.\n \* @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n

\*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun ULongArray.copyOfRange(fromIndex: Int, toIndex: Int): ULongArray {\n return ULongArray(storage.copyOfRange(fromIndex, toIndex))\n}\n\n/\*\*\n \* Returns a new array which is a copy of the specified range of the original array.\n \* @param fromIndex the start of the range (inclusive) to copy.\n \* @param toIndex the end of the range (exclusive) to copy.\n \* @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n \* @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n

\*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UByteArray.copyOfRange(fromIndex: Int, toIndex: Int): UByteArray {\n return UByteArray(storage.copyOfRange(fromIndex, toIndex))\n}\n\n/\*\*\n \* Returns a new array which is a copy of the specified range of the original array.\n \* @param fromIndex the start of the range (inclusive) to copy.\n

```

@param toIndex the end of the range (exclusive) to copy.\n * \n * @throws IndexOutOfBoundsException if
[fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n * @throws
IllegalArgumentException if [fromIndex] is greater than [toIndex].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.copyOfRange(fromIndex: Int, toIndex: Int): UShortArray {\n    return
UShortArray(storage.copyOfRange(fromIndex, toIndex))\n}\n\n/**\n * Fills this array or its subrange with the
specified [element] value.\n * \n * @param fromIndex the start of the range (inclusive) to fill, 0 by default.\n *
@param toIndex the end of the range (exclusive) to fill, size of this array by default.\n * \n * @throws
IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n *
@throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.fill(element: UInt, fromIndex:
Int = 0, toIndex: Int = size): Unit {\n    storage.fill(element.toInt(), fromIndex, toIndex)\n}\n\n/**\n * Fills this array
or its subrange with the specified [element] value.\n * \n * @param fromIndex the start of the range (inclusive) to
fill, 0 by default.\n * @param toIndex the end of the range (exclusive) to fill, size of this array by default.\n *
@param toIndex the end of the range (exclusive) to fill, size of this array by default.\n * \n * @throws
IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than the size of this
array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.fill(element: ULong,
fromIndex: Int = 0, toIndex: Int = size): Unit {\n    storage.fill(element.toLong(), fromIndex, toIndex)\n}\n\n/**\n *
Fills this array or its subrange with the specified [element] value.\n * \n * @param fromIndex the start of the range
(inclusive) to fill, 0 by default.\n * @param toIndex the end of the range (exclusive) to fill, size of this array by
default.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than
the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.fill(element: UByte,
fromIndex: Int = 0, toIndex: Int = size): Unit {\n    storage.fill(element.toByte(), fromIndex, toIndex)\n}\n\n/**\n *
Fills this array or its subrange with the specified [element] value.\n * \n * @param fromIndex the start of the range
(inclusive) to fill, 0 by default.\n * @param toIndex the end of the range (exclusive) to fill, size of this array by
default.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than
the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.fill(element: UShort,
fromIndex: Int = 0, toIndex: Int = size): Unit {\n    storage.fill(element.toShort(), fromIndex, toIndex)\n}\n\n/**\n *
Returns the range of valid indices for the array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic inline val UIntArray.indices: IntRange\n    get()
= storage.indices\n\n/**\n * Returns the range of valid indices for the array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic inline val ULongArray.indices: IntRange\n
    get() = storage.indices\n\n/**\n * Returns the range of valid indices for the array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic inline val UByteArray.indices: IntRange\n
    get() = storage.indices\n\n/**\n * Returns the range of valid indices for the array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic inline val UShortArray.indices: IntRange\n
    get() = storage.indices\n\n/**\n * Returns the last valid index for the array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic inline val UIntArray.lastIndex: Int\n    get() =
storage.lastIndex\n\n/**\n * Returns the last valid index for the array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic inline val ULongArray.lastIndex: Int\n    get() =
storage.lastIndex\n\n/**\n * Returns the last valid index for the array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic inline val UByteArray.lastIndex: Int\n    get() =
storage.lastIndex\n\n/**\n * Returns the last valid index for the array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic inline val UShortArray.lastIndex: Int\n    get() =
storage.lastIndex\n\n/**\n * Returns an array containing all elements of the original array and then the given
[element].\n *\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline

```

```

operator fun UIntArray.plus(element: UInt): UIntArray {
    return UIntArray(storage +
        element.toInt())
}
/**
 * Returns an array containing all elements of the original array and then the given
 * [element].
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline
operator fun ULongArray.plus(element: ULong): ULongArray {
    return ULongArray(storage +
        element.toLong())
}
/**
 * Returns an array containing all elements of the original array and then the given
 * [element].
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline
operator fun UByteArray.plus(element: UByte): UByteArray {
    return UByteArray(storage +
        element.toByte())
}
/**
 * Returns an array containing all elements of the original array and then the given
 * [element].
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline
operator fun UShortArray.plus(element: UShort): UShortArray {
    return UShortArray(storage +
        element.toShort())
}
/**
 * Returns an array containing all elements of the original array and then all elements
 * of the given [elements] collection.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
public operator
fun UIntArray.plus(elements: Collection<UInt>): UIntArray {
    var index = size
    val result =
        storage.copyOf(size + elements.size)
    for (element in elements) result[index++] = element.toInt()
    return
        UIntArray(result)
}
/**
 * Returns an array containing all elements of the original array and then all elements
 * of the given [elements] collection.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
public operator
fun ULongArray.plus(elements: Collection<ULong>): ULongArray {
    var index = size
    val result =
        storage.copyOf(size + elements.size)
    for (element in elements) result[index++] = element.toLong()
    return
        ULongArray(result)
}
/**
 * Returns an array containing all elements of the original array and then all
 * elements of the given [elements] collection.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
public operator
fun UByteArray.plus(elements: Collection<UByte>): UByteArray {
    var index = size
    val result =
        storage.copyOf(size + elements.size)
    for (element in elements) result[index++] = element.toByte()
    return
        UByteArray(result)
}
/**
 * Returns an array containing all elements of the original array and then all elements
 * of the given [elements] collection.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
public operator
fun UShortArray.plus(elements: Collection<UShort>): UShortArray {
    var index = size
    val result =
        storage.copyOf(size + elements.size)
    for (element in elements) result[index++] = element.toShort()
    return
        UShortArray(result)
}
/**
 * Returns an array containing all elements of the original array and then all
 * elements of the given [elements] array.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline operator fun
UIntArray.plus(elements: UIntArray): UIntArray {
    return UIntArray(storage + elements.storage)
}
/**
 * Returns an array containing all elements of the original array and then all elements
 * of the given [elements] array.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline operator fun
ULongArray.plus(elements: ULongArray): ULongArray {
    return ULongArray(storage +
        elements.storage)
}
/**
 * Returns an array containing all elements of the original array and then all elements
 * of the given [elements] array.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline operator fun
UByteArray.plus(elements: UByteArray): UByteArray {
    return UByteArray(storage +
        elements.storage)
}
/**
 * Returns an array containing all elements of the original array and then all elements
 * of the given [elements] array.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline operator fun
UShortArray.plus(elements: UShortArray): UShortArray {
    return UShortArray(storage +
        elements.storage)
}
/**
 * Sorts the array in-place.
 */
@sample
samples.collections.Arrays.Sorting.sortArray
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
public fun
UIntArray.sort(): Unit {
    if (size > 1) sortArray(this, 0, size)
}
/**
 * Sorts the array in-place.
 */
@sample
samples.collections.Arrays.Sorting.sortArray
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
public fun
ULongArray.sort(): Unit {
    if (size > 1)
        sortArray(this, 0, size)
}
/**
 * Sorts the array in-place.
 */
@sample
samples.collections.Arrays.Sorting.sortArray
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
public

```

```

fun UByteArray.sort(): Unit { \n  if (size > 1) sortArray(this, 0, size)\n}\n\n/**\n * Sorts the array in-place.\n * \n *
@sample samples.collections.Arrays.Sorting.sortArray\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.sort(): Unit { \n  if (size > 1)
sortArray(this, 0, size)\n}\n\n/**\n * Sorts a range in the array in-place.\n * \n * @param fromIndex the start of the
range (inclusive) to sort, 0 by default.\n * @param toIndex the end of the range (exclusive) to sort, size of this array
by default.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater
than the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n * \n *
@sample samples.collections.Arrays.Sorting.sortRangeOfArray\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.sort(fromIndex: Int = 0, toIndex:
Int = size): Unit { \n  AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n  sortArray(this, fromIndex,
toIndex)\n}\n\n/**\n * Sorts a range in the array in-place.\n * \n * @param fromIndex the start of the range
(inclusive) to sort, 0 by default.\n * @param toIndex the end of the range (exclusive) to sort, size of this array by
default.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater than
the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n * \n *
@sample samples.collections.Arrays.Sorting.sortRangeOfArray\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.sort(fromIndex: Int = 0,
toIndex: Int = size): Unit { \n  AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n  sortArray(this,
fromIndex, toIndex)\n}\n\n/**\n * Sorts a range in the array in-place.\n * \n * @param fromIndex the start of the
range (inclusive) to sort, 0 by default.\n * @param toIndex the end of the range (exclusive) to sort, size of this array
by default.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater
than the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n * \n *
@sample samples.collections.Arrays.Sorting.sortRangeOfArray\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.sort(fromIndex: Int = 0,
toIndex: Int = size): Unit { \n  AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n  sortArray(this,
fromIndex, toIndex)\n}\n\n/**\n * Sorts a range in the array in-place.\n * \n * @param fromIndex the start of the
range (inclusive) to sort, 0 by default.\n * @param toIndex the end of the range (exclusive) to sort, size of this array
by default.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is greater
than the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n * \n *
@sample samples.collections.Arrays.Sorting.sortRangeOfArray\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.sort(fromIndex: Int = 0,
toIndex: Int = size): Unit { \n  AbstractList.checkRangeIndexes(fromIndex, toIndex, size)\n  sortArray(this,
fromIndex, toIndex)\n}\n\n/**\n * Sorts elements of the array in the specified range in-place.\n * The elements are
sorted descending according to their natural sort order.\n * \n * @param fromIndex the start of the range (inclusive)
to sort.\n * @param toIndex the end of the range (exclusive) to sort.\n * \n * @throws IndexOutOfBoundsException
if [fromIndex] is less than zero or [toIndex] is greater than the size of this array.\n * @throws
IllegalArgumentException if [fromIndex] is greater than [toIndex].\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.sortDescending(fromIndex: Int,
toIndex: Int): Unit { \n  sort(fromIndex, toIndex)\n  reverse(fromIndex, toIndex)\n}\n\n/**\n * Sorts elements of
the array in the specified range in-place.\n * The elements are sorted descending according to their natural sort
order.\n * \n * @param fromIndex the start of the range (inclusive) to sort.\n * @param toIndex the end of the range
(exclusive) to sort.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is
greater than the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.sortDescending(fromIndex:
Int, toIndex: Int): Unit { \n  sort(fromIndex, toIndex)\n  reverse(fromIndex, toIndex)\n}\n\n/**\n * Sorts elements
of the array in the specified range in-place.\n * The elements are sorted descending according to their natural sort
order.\n * \n * @param fromIndex the start of the range (inclusive) to sort.\n * @param toIndex the end of the range
(exclusive) to sort.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is
greater than the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n

```

```

*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.sortDescending(fromIndex:
Int, toIndex: Int): Unit {\n    sort(fromIndex, toIndex)\n    reverse(fromIndex, toIndex)\n}\n\n/**\n * Sorts elements
of the array in the specified range in-place.\n * The elements are sorted descending according to their natural sort
order.\n * \n * @param fromIndex the start of the range (inclusive) to sort.\n * @param toIndex the end of the range
(exclusive) to sort.\n * \n * @throws IndexOutOfBoundsException if [fromIndex] is less than zero or [toIndex] is
greater than the size of this array.\n * @throws IllegalArgumentException if [fromIndex] is greater than [toIndex].\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.sortDescending(fromIndex:
Int, toIndex: Int): Unit {\n    sort(fromIndex, toIndex)\n    reverse(fromIndex, toIndex)\n}\n\n/**\n * Returns an
array of type [ByteArray], which is a copy of this array where each element is a signed reinterpretation\n * of the
corresponding element of this array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.toByteArray(): ByteArray {\n    return storage.copyOf()\n}\n\n/**\n * Returns an array of type
[IntArray], which is a copy of this array where each element is a signed reinterpretation\n * of the corresponding
element of this array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.toIntArray(): IntArray {\n    return storage.copyOf()\n}\n\n/**\n * Returns an array of type [LongArray],
which is a copy of this array where each element is a signed reinterpretation\n * of the corresponding element of this
array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.toLongArray(): LongArray {\n    return storage.copyOf()\n}\n\n/**\n * Returns an array of type
[ShortArray], which is a copy of this array where each element is a signed reinterpretation\n * of the corresponding
element of this array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.toShortArray(): ShortArray {\n    return storage.copyOf()\n}\n\n/**\n * Returns a *typed* object array
containing all of the elements of this primitive array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.toTypedArray(): Array<UInt>
{\n    return Array(size) { index -> this[index] }\n}\n\n/**\n * Returns a *typed* object array containing all of the
elements of this primitive array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun
ULongArray.toTypedArray(): Array<ULong> {\n    return Array(size) { index -> this[index] }\n}\n\n/**\n *
Returns a *typed* object array containing all of the elements of this primitive array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.toTypedArray():
Array<UByte> {\n    return Array(size) { index -> this[index] }\n}\n\n/**\n * Returns a *typed* object array
containing all of the elements of this primitive array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.toTypedArray():
Array<UShort> {\n    return Array(size) { index -> this[index] }\n}\n\n/**\n * Returns an array of UByte containing
all of the elements of this generic array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun
Array<out UByte>.toUByteArray(): UByteArray {\n    return UByteArray(size) { index -> this[index] }\n}\n\n/**\n
* Returns an array of type [UByteArray], which is a copy of this array where each element is an unsigned
reinterpretation\n * of the corresponding element of this array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ByteArray.toUByteArray(): UByteArray {\n    return UByteArray(this.copyOf())\n}\n\n/**\n * Returns an array of
UInt containing all of the elements of this generic array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun Array<out UInt>.toUIntArray(): UIntArray
{\n    return UIntArray(size) { index -> this[index] }\n}\n\n/**\n * Returns an array of type [UIntArray], which is a
copy of this array where each element is an unsigned reinterpretation\n * of the corresponding element of this
array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
IntArray.toUIntArray(): UIntArray {\n    return UIntArray(this.copyOf())\n}\n\n/**\n * Returns an array of ULong
containing all of the elements of this generic array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun Array<out ULong>.toULongArray():

```

```

ULongArray {\n  return ULongArray(size) { index -> this[index] }\n}\n\n/**\n * Returns an array of type
[ULongArray], which is a copy of this array where each element is an unsigned reinterpretation\n * of the
corresponding element of this array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
LongArray.toULongArray(): ULongArray {\n  return ULongArray(this.copyOf())\n}\n\n/**\n * Returns an array
of UShort containing all of the elements of this generic array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun Array<out UShort>.toUShortArray():
UShortArray {\n  return UShortArray(size) { index -> this[index] }\n}\n\n/**\n * Returns an array of type
[UShortArray], which is a copy of this array where each element is an unsigned reinterpretation\n * of the
corresponding element of this array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ShortArray.toUShortArray(): UShortArray {\n  return UShortArray(this.copyOf())\n}\n\n/**\n * Returns a [Map]
where keys are elements from the given array and values are\n * produced by the [valueSelector] function applied to
each element.\n * \n * If any two elements are equal, the last one gets added to the map.\n * \n * The returned map
preserves the entry iteration order of the original array.\n * \n * @sample
samples.collections.Collections.Transformations.associateWith\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <V>
UIntArray.associateWith(valueSelector: (UInt) -> V): Map<UInt, V> {\n  val result = LinkedHashMap<UInt,
V>(mapCapacity(size).coerceAtLeast(16))\n  return associateWithTo(result, valueSelector)\n}\n\n/**\n * Returns a
[Map] where keys are elements from the given array and values are\n * produced by the [valueSelector] function
applied to each element.\n * \n * If any two elements are equal, the last one gets added to the map.\n * \n * The
returned map preserves the entry iteration order of the original array.\n * \n * @sample
samples.collections.Collections.Transformations.associateWith\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <V>
ULongArray.associateWith(valueSelector: (ULong) -> V): Map<ULong, V> {\n  val result =
LinkedHashMap<ULong, V>(mapCapacity(size).coerceAtLeast(16))\n  return associateWithTo(result,
valueSelector)\n}\n\n/**\n * Returns a [Map] where keys are elements from the given array and values are\n *
produced by the [valueSelector] function applied to each element.\n * \n * If any two elements are equal, the last one
gets added to the map.\n * \n * The returned map preserves the entry iteration order of the original array.\n * \n *
@sample samples.collections.Collections.Transformations.associateWith\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <V>
UByteArray.associateWith(valueSelector: (UByte) -> V): Map<UByte, V> {\n  val result =
LinkedHashMap<UByte, V>(mapCapacity(size).coerceAtLeast(16))\n  return associateWithTo(result,
valueSelector)\n}\n\n/**\n * Returns a [Map] where keys are elements from the given array and values are\n *
produced by the [valueSelector] function applied to each element.\n * \n * If any two elements are equal, the last one
gets added to the map.\n * \n * The returned map preserves the entry iteration order of the original array.\n * \n *
@sample samples.collections.Collections.Transformations.associateWith\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <V>
UShortArray.associateWith(valueSelector: (UShort) -> V): Map<UShort, V> {\n  val result =
LinkedHashMap<UShort, V>(mapCapacity(size).coerceAtLeast(16))\n  return associateWithTo(result,
valueSelector)\n}\n\n/**\n * Populates and returns the [destination] mutable map with key-value pairs for each
element of the given array,\n * where key is the element itself and value is provided by the [valueSelector] function
applied to that key.\n * \n * If any two elements are equal, the last one overwrites the former value in the map.\n * \n
* @sample samples.collections.Collections.Transformations.associateWithTo\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <V, M :
MutableMap<in UInt, in V>> UIntArray.associateWithTo(destination: M, valueSelector: (UInt) -> V): M {\n  for
(element in this) {\n    destination.put(element, valueSelector(element))\n  }\n  return destination\n}\n\n/**\n *
Populates and returns the [destination] mutable map with key-value pairs for each element of the given array,\n *

```

where key is the element itself and value is provided by the [valueSelector] function applied to that key.\n \* \n \* If any two elements are equal, the last one overwrites the former value in the map.\n \* \n \* @sample samples.collections.Collections.Transformations.associateWithTo\n

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <V, M : MutableMap<in ULong, in V>> ULongArray.associateWithTo(destination: M, valueSelector: (ULong) -> V): M {\n    for (element in this) {\n        destination.put(element, valueSelector(element))\n    }\n    return destination\n}\n\n/**\n * Populates and returns the [destination] mutable map with key-value pairs for each element of the given array,\n * where key is the element itself and value is provided by the [valueSelector] function applied to that key.\n * \n * If any two elements are equal, the last one overwrites the former value in the map.\n * \n * @sample samples.collections.Collections.Transformations.associateWithTo\n
```

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <V, M : MutableMap<in UByte, in V>> UByteArray.associateWithTo(destination: M, valueSelector: (UByte) -> V): M {\n    for (element in this) {\n        destination.put(element, valueSelector(element))\n    }\n    return destination\n}\n\n/**\n * Populates and returns the [destination] mutable map with key-value pairs for each element of the given array,\n * where key is the element itself and value is provided by the [valueSelector] function applied to that key.\n * \n * If any two elements are equal, the last one overwrites the former value in the map.\n * \n * @sample samples.collections.Collections.Transformations.associateWithTo\n
```

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <V, M : MutableMap<in UShort, in V>> UShortArray.associateWithTo(destination: M, valueSelector: (UShort) -> V): M {\n    for (element in this) {\n        destination.put(element, valueSelector(element))\n    }\n    return destination\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n * \n * @sample samples.collections.Collections.Transformations.flatMap\n
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R> UIntArray.flatMap(transform: (UInt) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n * \n * @sample samples.collections.Collections.Transformations.flatMap\n
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R> ULongArray.flatMap(transform: (ULong) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n * \n * @sample samples.collections.Collections.Transformations.flatMap\n
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R> UByteArray.flatMap(transform: (UByte) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element of original array.\n * \n * @sample samples.collections.Collections.Transformations.flatMap\n
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R> UShortArray.flatMap(transform: (UShort) -> Iterable<R>): List<R> {\n    return flatMapTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element\n * and its index in the original array.\n * \n * @sample samples.collections.Collections.Transformations.flatMapIndexed\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolutionByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R> UIntArray.flatMapIndexed(transform: (index: Int, UInt) -> Iterable<R>): List<R> {\n    return flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n/**\n * Returns a single list of all elements yielded from results of [transform] function being invoked on each element\n * and its index in the original array.\n * \n * @sample samples.collections.Collections.Transformations.flatMapIndexed\n
```

```

@sample samples.collections.Collections.Transformations.flatMapIndexed\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
ULongArray.flatMapIndexed(transform: (index: Int, ULong) -> Iterable<R>): List<R> {\n return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n**\n * Returns a single list of all elements yielded from
results of [transform] function being invoked on each element\n * and its index in the original array.\n * \n *
@sample samples.collections.Collections.Transformations.flatMapIndexed\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UByteArray.flatMapIndexed(transform: (index: Int, UByte) -> Iterable<R>): List<R> {\n return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n**\n * Returns a single list of all elements yielded from
results of [transform] function being invoked on each element\n * and its index in the original array.\n * \n *
@sample samples.collections.Collections.Transformations.flatMapIndexed\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UShortArray.flatMapIndexed(transform: (index: Int, UShort) -> Iterable<R>): List<R> {\n return
flatMapIndexedTo(ArrayList<R>(), transform)\n}\n\n**\n * Appends all elements yielded from results of
[transform] function being invoked on each element\n * and its index in the original array, to the given
[destination].\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> UIntArray.flatMapIndexedTo(destination: C, transform: (index: Int, UInt) ->
Iterable<R>): C {\n var index = 0\n for (element in this) {\n val list = transform(index++, element)\n
destination.addAll(list)\n }\n return destination\n}\n\n**\n * Appends all elements yielded from results of
[transform] function being invoked on each element\n * and its index in the original array, to the given
[destination].\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> ULongArray.flatMapIndexedTo(destination: C, transform: (index: Int, ULong) ->
Iterable<R>): C {\n var index = 0\n for (element in this) {\n val list = transform(index++, element)\n
destination.addAll(list)\n }\n return destination\n}\n\n**\n * Appends all elements yielded from results of
[transform] function being invoked on each element\n * and its index in the original array, to the given
[destination].\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> UByteArray.flatMapIndexedTo(destination: C, transform: (index: Int, UByte) ->
Iterable<R>): C {\n var index = 0\n for (element in this) {\n val list = transform(index++, element)\n
destination.addAll(list)\n }\n return destination\n}\n\n**\n * Appends all elements yielded from results of
[transform] function being invoked on each element\n * and its index in the original array, to the given
[destination].\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> UShortArray.flatMapIndexedTo(destination: C, transform: (index: Int, UShort) ->
Iterable<R>): C {\n var index = 0\n for (element in this) {\n val list = transform(index++, element)\n
destination.addAll(list)\n }\n return destination\n}\n\n**\n * Appends all elements yielded from results of
[transform] function being invoked on each element of original array, to the given [destination].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> UIntArray.flatMapTo(destination: C, transform: (UInt) -> Iterable<R>): C {\n for

```

```

(element in this) {\n    val list = transform(element)\n    destination.addAll(list)\n } \n return
destination\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each
element of original array, to the given [destination].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> ULongArray.flatMapTo(destination: C, transform: (ULong) -> Iterable<R>): C {\n for
(element in this) {\n    val list = transform(element)\n    destination.addAll(list)\n } \n return
destination\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each
element of original array, to the given [destination].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> UByteArray.flatMapTo(destination: C, transform: (UByte) -> Iterable<R>): C {\n for
(element in this) {\n    val list = transform(element)\n    destination.addAll(list)\n } \n return
destination\n}\n\n/**\n * Appends all elements yielded from results of [transform] function being invoked on each
element of original array, to the given [destination].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> UShortArray.flatMapTo(destination: C, transform: (UShort) -> Iterable<R>): C {\n for
(element in this) {\n    val list = transform(element)\n    destination.addAll(list)\n } \n return
destination\n}\n\n/**\n * Groups elements of the original array by the key returned by the given [keySelector]
function\n * applied to each element and returns a map where each group key is associated with a list of
corresponding elements.\n * \n * The returned map preserves the entry iteration order of the keys produced from the
original array.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K>
UIntArray.groupBy(keySelector: (UInt) -> K): Map<K, List<UInt>> {\n return groupByTo(LinkedHashMap<K,
MutableList<UInt>>(), keySelector)\n}\n\n/**\n * Groups elements of the original array by the key returned by the
given [keySelector] function\n * applied to each element and returns a map where each group key is associated with
a list of corresponding elements.\n * \n * The returned map preserves the entry iteration order of the keys produced
from the original array.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K>
ULongArray.groupBy(keySelector: (ULong) -> K): Map<K, List<ULong>> {\n return
groupByTo(LinkedHashMap<K, MutableList<ULong>>(), keySelector)\n}\n\n/**\n * Groups elements of the
original array by the key returned by the given [keySelector] function\n * applied to each element and returns a map
where each group key is associated with a list of corresponding elements.\n * \n * The returned map preserves the
entry iteration order of the keys produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupBy\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K>
UByteArray.groupBy(keySelector: (UByte) -> K): Map<K, List<UByte>> {\n return
groupByTo(LinkedHashMap<K, MutableList<UByte>>(), keySelector)\n}\n\n/**\n * Groups elements of the
original array by the key returned by the given [keySelector] function\n * applied to each element and returns a map
where each group key is associated with a list of corresponding elements.\n * \n * The returned map preserves the
entry iteration order of the keys produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupBy\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K>
UShortArray.groupBy(keySelector: (UShort) -> K): Map<K, List<UShort>> {\n return
groupByTo(LinkedHashMap<K, MutableList<UShort>>(), keySelector)\n}\n\n/**\n * Groups values returned by
the [valueTransform] function applied to each element of the original array\n * by the key returned by the given
[keySelector] function applied to the element\n * and returns a map where each group key is associated with a list of
corresponding values.\n * \n * The returned map preserves the entry iteration order of the keys produced from the
original array.\n * \n * @sample samples.collections.Collections.Transformations.groupByKeysAndValues\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K, V>

```

```

UIntArray.groupBy(keySelector: (UInt) -> K, valueTransform: (UInt) -> V): Map<K, List<V>> {\n  return
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups values
returned by the [valueTransform] function applied to each element of the original array\n * by the key returned by
the given [keySelector] function applied to the element\n * and returns a map where each group key is associated
with a list of corresponding values.\n * \n * The returned map preserves the entry iteration order of the keys
produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupByKeyAndValues\n
*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K, V>
ULongArray.groupBy(keySelector: (ULong) -> K, valueTransform: (ULong) -> V): Map<K, List<V>> {\n  return
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups values
returned by the [valueTransform] function applied to each element of the original array\n * by the key returned by
the given [keySelector] function applied to the element\n * and returns a map where each group key is associated
with a list of corresponding values.\n * \n * The returned map preserves the entry iteration order of the keys
produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupByKeyAndValues\n
*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K, V>
UByteArray.groupBy(keySelector: (UByte) -> K, valueTransform: (UByte) -> V): Map<K, List<V>> {\n  return
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups values
returned by the [valueTransform] function applied to each element of the original array\n * by the key returned by
the given [keySelector] function applied to the element\n * and returns a map where each group key is associated
with a list of corresponding values.\n * \n * The returned map preserves the entry iteration order of the keys
produced from the original array.\n * \n * @sample
samples.collections.Collections.Transformations.groupByKeyAndValues\n
*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K, V>
UShortArray.groupBy(keySelector: (UShort) -> K, valueTransform: (UShort) -> V): Map<K, List<V>> {\n  return
groupByTo(LinkedHashMap<K, MutableList<V>>(), keySelector, valueTransform)\n}\n\n/**\n * Groups elements
of the original array by the key returned by the given [keySelector] function\n * applied to each element and puts to
the [destination] map each group key associated with a list of corresponding elements.\n * \n * @return The
[destination] map.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n
*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K, M :
MutableMap<in K, MutableList<UInt>>>> UIntArray.groupByTo(destination: M, keySelector: (UInt) -> K): M {\n
for (element in this) {\n    val key = keySelector(element)\n    val list = destination.getOrPut(key) {\n
ArrayList<UInt>()\n    } list.add(element)\n    }\n    return destination\n}\n\n/**\n * Groups elements of the
original array by the key returned by the given [keySelector] function\n * applied to each element and puts to the
[destination] map each group key associated with a list of corresponding elements.\n * \n * @return The
[destination] map.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n
*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K, M :
MutableMap<in K, MutableList<ULong>>>> ULongArray.groupByTo(destination: M, keySelector: (ULong) -> K):
M {\n for (element in this) {\n    val key = keySelector(element)\n    val list = destination.getOrPut(key) {\n
ArrayList<ULong>()\n    } list.add(element)\n    }\n    return destination\n}\n\n/**\n * Groups elements of the
original array by the key returned by the given [keySelector] function\n * applied to each element and puts to the
[destination] map each group key associated with a list of corresponding elements.\n * \n * @return The
[destination] map.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n
*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K, M :
MutableMap<in K, MutableList<UByte>>>> UByteArray.groupByTo(destination: M, keySelector: (UByte) -> K):
M {\n for (element in this) {\n    val key = keySelector(element)\n    val list = destination.getOrPut(key) {\n
ArrayList<UByte>()\n    } list.add(element)\n    }\n    return destination\n}\n\n/**\n * Groups elements of the
original array by the key returned by the given [keySelector] function\n * applied to each element and puts to the

```

```

[destination] map each group key associated with a list of corresponding elements.\n * \n * @return The
[destination] map.\n * \n * @sample samples.collections.Collections.Transformations.groupBy\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K, M :
MutableMap<in K, MutableList<UShort>>> UShortArray.groupByTo(destination: M, keySelector: (UShort) -> K):
M {\n for (element in this) {\n val key = keySelector(element)\n val list = destination.getOrPut(key) {
ArrayList<UShort>() }\n list.add(element)\n }\n return destination}\n\n/**\n * Groups values returned by
the [valueTransform] function applied to each element of the original array\n * by the key returned by the given
[keySelector] function applied to the element\n * and puts to the [destination] map each group key associated with a
list of corresponding values.\n * \n * @return The [destination] map.\n * \n * @sample
samples.collections.Collections.Transformations.groupByKeysAndValues\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K, V,
M : MutableMap<in K, MutableList<V>>> UIntArray.groupByTo(destination: M, keySelector: (UInt) -> K,
valueTransform: (UInt) -> V): M {\n for (element in this) {\n val key = keySelector(element)\n val list =
destination.getOrPut(key) { ArrayList<V>() }\n list.add(valueTransform(element))\n }\n return
destination}\n\n/**\n * Groups values returned by the [valueTransform] function applied to each element of the
original array\n * by the key returned by the given [keySelector] function applied to the element\n * and puts to the
[destination] map each group key associated with a list of corresponding values.\n * \n * @return The [destination]
map.\n * \n * @sample samples.collections.Collections.Transformations.groupByKeysAndValues\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K, V,
M : MutableMap<in K, MutableList<V>>> ULongArray.groupByTo(destination: M, keySelector: (ULong) -> K,
valueTransform: (ULong) -> V): M {\n for (element in this) {\n val key = keySelector(element)\n val list
= destination.getOrPut(key) { ArrayList<V>() }\n list.add(valueTransform(element))\n }\n return
destination}\n\n/**\n * Groups values returned by the [valueTransform] function applied to each element of the
original array\n * by the key returned by the given [keySelector] function applied to the element\n * and puts to the
[destination] map each group key associated with a list of corresponding values.\n * \n * @return The [destination]
map.\n * \n * @sample samples.collections.Collections.Transformations.groupByKeysAndValues\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K, V,
M : MutableMap<in K, MutableList<V>>> UByteArray.groupByTo(destination: M, keySelector: (UByte) -> K,
valueTransform: (UByte) -> V): M {\n for (element in this) {\n val key = keySelector(element)\n val list
= destination.getOrPut(key) { ArrayList<V>() }\n list.add(valueTransform(element))\n }\n return
destination}\n\n/**\n * Groups values returned by the [valueTransform] function applied to each element of the
original array\n * by the key returned by the given [keySelector] function applied to the element\n * and puts to the
[destination] map each group key associated with a list of corresponding values.\n * \n * @return The [destination]
map.\n * \n * @sample samples.collections.Collections.Transformations.groupByKeysAndValues\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <K, V,
M : MutableMap<in K, MutableList<V>>> UShortArray.groupByTo(destination: M, keySelector: (UShort) -> K,
valueTransform: (UShort) -> V): M {\n for (element in this) {\n val key = keySelector(element)\n val list
= destination.getOrPut(key) { ArrayList<V>() }\n list.add(valueTransform(element))\n }\n return
destination}\n\n/**\n * Returns a list containing the results of applying the given [transform] function\n * to each
element in the original array.\n * \n * @sample samples.collections.Collections.Transformations.map\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UIntArray.map(transform: (UInt) -> R): List<R> {\n return mapTo(ArrayList<R>(size), transform)\n}\n\n/**\n *
Returns a list containing the results of applying the given [transform] function\n * to each element in the original
array.\n * \n * @sample samples.collections.Collections.Transformations.map\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
ULongArray.map(transform: (ULong) -> R): List<R> {\n return mapTo(ArrayList<R>(size),
transform)\n}\n\n/**\n * Returns a list containing the results of applying the given [transform] function\n * to each
element in the original array.\n * \n * @sample samples.collections.Collections.Transformations.map\n

```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UByteArray.map(transform: (UByte) -> R): List<R> {\n    return mapTo(ArrayList<R>(size),
transform)\n}\n\n/**\n * Returns a list containing the results of applying the given [transform] function\n * to each
element in the original array.\n * \n * @sample samples.collections.Collections.Transformations.map\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UShortArray.map(transform: (UShort) -> R): List<R> {\n    return mapTo(ArrayList<R>(size),
transform)\n}\n\n/**\n * Returns a list containing the results of applying the given [transform] function\n * to each
element and its index in the original array.\n * @param [transform] function that takes the index of an element and
the element itself\n * and returns the result of the transform applied to the element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UIntArray.mapIndexed(transform: (index: Int, UInt) -> R): List<R> {\n    return
mapIndexedTo(ArrayList<R>(size), transform)\n}\n\n/**\n * Returns a list containing the results of applying the
given [transform] function\n * to each element and its index in the original array.\n * @param [transform] function
that takes the index of an element and the element itself\n * and returns the result of the transform applied to the
element.\n *\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline
fun <R> ULongArray.mapIndexed(transform: (index: Int, ULong) -> R): List<R> {\n    return
mapIndexedTo(ArrayList<R>(size), transform)\n}\n\n/**\n * Returns a list containing the results of applying the
given [transform] function\n * to each element and its index in the original array.\n * @param [transform] function
that takes the index of an element and the element itself\n * and returns the result of the transform applied to the
element.\n *\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline
fun <R> UByteArray.mapIndexed(transform: (index: Int, UByte) -> R): List<R> {\n    return
mapIndexedTo(ArrayList<R>(size), transform)\n}\n\n/**\n * Returns a list containing the results of applying the
given [transform] function\n * to each element and its index in the original array.\n * @param [transform] function
that takes the index of an element and the element itself\n * and returns the result of the transform applied to the
element.\n *\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline
fun <R> UShortArray.mapIndexed(transform: (index: Int, UShort) -> R): List<R> {\n    return
mapIndexedTo(ArrayList<R>(size), transform)\n}\n\n/**\n * Applies the given [transform] function to each
element and its index in the original array\n * and appends the results to the given [destination].\n * @param
[transform] function that takes the index of an element and the element itself\n * and returns the result of the
transform applied to the element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> UIntArray.mapIndexedTo(destination: C, transform: (index: Int, UInt) -> R): C {\n    var
index = 0\n    for (item in this)\n        destination.add(transform(index++, item))\n    return destination\n}\n\n/**\n * Applies the given [transform] function to each element and its index in the original array\n * and appends the results
to the given [destination].\n * @param [transform] function that takes the index of an element and the element
itself\n * and returns the result of the transform applied to the element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> ULongArray.mapIndexedTo(destination: C, transform: (index: Int, ULong) -> R): C {\n    var
index = 0\n    for (item in this)\n        destination.add(transform(index++, item))\n    return
destination\n}\n\n/**\n * Applies the given [transform] function to each element and its index in the original array\n
* and appends the results to the given [destination].\n * @param [transform] function that takes the index of an
element and the element itself\n * and returns the result of the transform applied to the element.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> UByteArray.mapIndexedTo(destination: C, transform: (index: Int, UByte) -> R): C {\n    var
index = 0\n    for (item in this)\n        destination.add(transform(index++, item))\n    return
destination\n}\n\n/**\n * Applies the given [transform] function to each element and its index in the original array\n
* and appends the results to the given [destination].\n * @param [transform] function that takes the index of an
element and the element itself\n * and returns the result of the transform applied to the element.\n

```

```

*^@SinceKotlin("1.3")@ExperimentalUnsignedTypes@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> UShortArray.mapIndexedTo(destination: C, transform: (index: Int, UShort) -> R): C {\n
var index = 0\n for (item in this)\n destination.add(transform(index++, item))\n return
destination\n}\n\n/**\n * Applies the given [transform] function to each element of the original array\n * and
appends the results to the given [destination].\n
*^@SinceKotlin("1.3")@ExperimentalUnsignedTypes@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> UIntArray.mapTo(destination: C, transform: (UInt) -> R): C {\n for (item in this)\n
destination.add(transform(item))\n return destination\n}\n\n/**\n * Applies the given [transform] function to each
element of the original array\n * and appends the results to the given [destination].\n
*^@SinceKotlin("1.3")@ExperimentalUnsignedTypes@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> ULongArray.mapTo(destination: C, transform: (ULong) -> R): C {\n for (item in
this)\n destination.add(transform(item))\n return destination\n}\n\n/**\n * Applies the given [transform]
function to each element of the original array\n * and appends the results to the given [destination].\n
*^@SinceKotlin("1.3")@ExperimentalUnsignedTypes@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> UByteArray.mapTo(destination: C, transform: (UByte) -> R): C {\n for (item in this)\n
destination.add(transform(item))\n return destination\n}\n\n/**\n * Applies the given [transform] function to
each element of the original array\n * and appends the results to the given [destination].\n
*^@SinceKotlin("1.3")@ExperimentalUnsignedTypes@kotlin.internal.InlineOnly\npublic inline fun <R, C :
MutableCollection<in R>> UShortArray.mapTo(destination: C, transform: (UShort) -> R): C {\n for (item in
this)\n destination.add(transform(item))\n return destination\n}\n\n/**\n * Returns a lazy [Iterable] that wraps
each element of the original array\n * into an [IndexedValue] containing the index of that element and the element
itself.\n *^@SinceKotlin("1.3")@ExperimentalUnsignedTypes\npublic fun UIntArray.withIndex():
Iterable<IndexedValue<UInt>> {\n return IndexingIterable { iterator() }\n}\n\n/**\n * Returns a lazy [Iterable]
that wraps each element of the original array\n * into an [IndexedValue] containing the index of that element and the
element itself.\n *^@SinceKotlin("1.3")@ExperimentalUnsignedTypes\npublic fun ULongArray.withIndex():
Iterable<IndexedValue<ULong>> {\n return IndexingIterable { iterator() }\n}\n\n/**\n * Returns a lazy [Iterable]
that wraps each element of the original array\n * into an [IndexedValue] containing the index of that element and the
element itself.\n *^@SinceKotlin("1.3")@ExperimentalUnsignedTypes\npublic fun UByteArray.withIndex():
Iterable<IndexedValue<UByte>> {\n return IndexingIterable { iterator() }\n}\n\n/**\n * Returns a lazy [Iterable]
that wraps each element of the original array\n * into an [IndexedValue] containing the index of that element and the
element itself.\n *^@SinceKotlin("1.3")@ExperimentalUnsignedTypes\npublic fun UShortArray.withIndex():
Iterable<IndexedValue<UShort>> {\n return IndexingIterable { iterator() }\n}\n\n/**\n * Returns `true` if all
elements match the given [predicate].\n * \n * @sample samples.collections.Collections.Aggregates.all\n
*^@SinceKotlin("1.3")@ExperimentalUnsignedTypes@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.all(predicate: (UInt) -> Boolean): Boolean {\n for (element in this) if (!predicate(element)) return
false\n return true\n}\n\n/**\n * Returns `true` if all elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.all\n
*^@SinceKotlin("1.3")@ExperimentalUnsignedTypes@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.all(predicate: (ULong) -> Boolean): Boolean {\n for (element in this) if (!predicate(element)) return
false\n return true\n}\n\n/**\n * Returns `true` if all elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.all\n
*^@SinceKotlin("1.3")@ExperimentalUnsignedTypes@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.all(predicate: (UByte) -> Boolean): Boolean {\n for (element in this) if (!predicate(element)) return
false\n return true\n}\n\n/**\n * Returns `true` if all elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.all\n
*^@SinceKotlin("1.3")@ExperimentalUnsignedTypes@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.all(predicate: (UShort) -> Boolean): Boolean {\n for (element in this) if (!predicate(element)) return
false\n return true\n}\n\n/**\n * Returns `true` if array has at least one element.\n * \n * @sample

```

```

samples.collections.Collections.Aggregates.any\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.any(): Boolean {\n    return storage.any()\n}\n\n/**\n * Returns `true` if array has at least one element.\n
 * \n * @sample samples.collections.Collections.Aggregates.any\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.any(): Boolean {\n    return storage.any()\n}\n\n/**\n * Returns `true` if array has at least one
element.\n * \n * @sample samples.collections.Collections.Aggregates.any\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.any(): Boolean {\n    return storage.any()\n}\n\n/**\n * Returns `true` if array has at least one
element.\n * \n * @sample samples.collections.Collections.Aggregates.any\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.any(): Boolean {\n    return storage.any()\n}\n\n/**\n * Returns `true` if at least one element matches
the given [predicate].\n * \n * @sample samples.collections.Collections.Aggregates.anyWithPredicate\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.any(predicate: (UInt) -> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return true\n
return false\n}\n\n/**\n * Returns `true` if at least one element matches the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.anyWithPredicate\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.any(predicate: (ULong) -> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return
true\n    return false\n}\n\n/**\n * Returns `true` if at least one element matches the given [predicate].\n * \n *
@sample samples.collections.Collections.Aggregates.anyWithPredicate\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.any(predicate: (UByte) -> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return
true\n    return false\n}\n\n/**\n * Returns `true` if at least one element matches the given [predicate].\n * \n *
@sample samples.collections.Collections.Aggregates.anyWithPredicate\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.any(predicate: (UShort) -> Boolean): Boolean {\n    for (element in this) if (predicate(element)) return
true\n    return false\n}\n\n/**\n * Returns the number of elements matching the given [predicate].\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.count(predicate: (UInt) -> Boolean): Int {\n    var count = 0\n    for (element in this) if
(predicate(element)) ++count\n    return count\n}\n\n/**\n * Returns the number of elements matching the given
[predicate].\n * \n *\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic
inline fun ULongArray.count(predicate: (ULong) -> Boolean): Int {\n    var count = 0\n    for (element in this) if
(predicate(element)) ++count\n    return count\n}\n\n/**\n * Returns the number of elements matching the given
[predicate].\n * \n *\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic
inline fun UByteArray.count(predicate: (UByte) -> Boolean): Int {\n    var count = 0\n    for (element in this) if
(predicate(element)) ++count\n    return count\n}\n\n/**\n * Returns the number of elements matching the given
[predicate].\n * \n *\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic
inline fun UShortArray.count(predicate: (UShort) -> Boolean): Int {\n    var count = 0\n    for (element in this) if
(predicate(element)) ++count\n    return count\n}\n\n/**\n * Accumulates value starting with [initial] value and
applying [operation] from left to right\n * to current accumulator value and each element.\n * \n * Returns the
specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes current accumulator
value and an element, and calculates the next accumulator value.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UIntArray.fold(initial: R, operation: (acc: R, UInt) -> R): R {\n    var accumulator = initial\n    for (element in this)
accumulator = operation(accumulator, element)\n    return accumulator\n}\n\n/**\n * Accumulates value starting
with [initial] value and applying [operation] from left to right\n * to current accumulator value and each element.\n *
*\n * Returns the specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes

```

current accumulator value and an element, and calculates the next accumulator value.\n

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nULongArray.fold(initial: R, operation: (acc: R, ULong) -> R): R {\n    var accumulator = initial\n    for (element in\n        this) accumulator = operation(accumulator, element)\n    return accumulator\n}\n\n/**\n * Accumulates value\n * starting with [initial] value and applying [operation] from left to right\n * to current accumulator value and each\n * element.\n * Returns the specified [initial] value if the array is empty.\n * @param [operation] function that\n * takes current accumulator value and an element, and calculates the next accumulator value.\n */
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nUByteArray.fold(initial: R, operation: (acc: R, UByte) -> R): R {\n    var accumulator = initial\n    for (element in\n        this) accumulator = operation(accumulator, element)\n    return accumulator\n}\n\n/**\n * Accumulates value\n * starting with [initial] value and applying [operation] from left to right\n * to current accumulator value and each\n * element.\n * Returns the specified [initial] value if the array is empty.\n * @param [operation] function that\n * takes current accumulator value and an element, and calculates the next accumulator value.\n */
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nUShortArray.fold(initial: R, operation: (acc: R, UShort) -> R): R {\n    var accumulator = initial\n    for (element in\n        this) accumulator = operation(accumulator, element)\n    return accumulator\n}\n\n/**\n * Accumulates value\n * starting with [initial] value and applying [operation] from left to right\n * to current accumulator value and each\n * element with its index in the original array.\n * Returns the specified [initial] value if the array is empty.\n * @param [operation] function that\n * takes the index of an element, current accumulator value\n * and the element\n * itself, and calculates the next accumulator value.\n */
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nUIntArray.foldIndexed(initial: R, operation: (index: Int, acc: R, UInt) -> R): R {\n    var index = 0\n    var\n        accumulator = initial\n    for (element in this) accumulator = operation(index++, accumulator, element)\n    return\n        accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from left to\n * right\n * to current accumulator value and each element with its index in the original array.\n * Returns the\n * specified [initial] value if the array is empty.\n * @param [operation] function that\n * takes the index of an\n * element, current accumulator value\n * and the element itself, and calculates the next accumulator value.\n */
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nULongArray.foldIndexed(initial: R, operation: (index: Int, acc: R, ULong) -> R): R {\n    var index = 0\n    var\n        accumulator = initial\n    for (element in this) accumulator = operation(index++, accumulator, element)\n    return\n        accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from left to\n * right\n * to current accumulator value and each element with its index in the original array.\n * Returns the\n * specified [initial] value if the array is empty.\n * @param [operation] function that\n * takes the index of an\n * element, current accumulator value\n * and the element itself, and calculates the next accumulator value.\n */
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nUByteArray.foldIndexed(initial: R, operation: (index: Int, acc: R, UByte) -> R): R {\n    var index = 0\n    var\n        accumulator = initial\n    for (element in this) accumulator = operation(index++, accumulator, element)\n    return\n        accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from left to\n * right\n * to current accumulator value and each element with its index in the original array.\n * Returns the\n * specified [initial] value if the array is empty.\n * @param [operation] function that\n * takes the index of an\n * element, current accumulator value\n * and the element itself, and calculates the next accumulator value.\n */
```

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nUShortArray.foldIndexed(initial: R, operation: (index: Int, acc: R, UShort) -> R): R {\n    var index = 0\n    var\n        accumulator = initial\n    for (element in this) accumulator = operation(index++, accumulator, element)\n    return\n        accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from right to\n * left\n * to each element and current accumulator value.\n * Returns the specified [initial] value if the array is\n * empty.\n * @param [operation] function that\n * takes an element and current accumulator value, and calculates the\n * next accumulator value.\n */
```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UIntArray.foldRight(initial: R, operation: (UInt, acc: R) -> R): R {\n  var index = lastIndex\n  var accumulator =
initial\n  while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n  return
accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from right to
left\n * to each element and current accumulator value.\n * \n * Returns the specified [initial] value if the array is
empty.\n * \n * @param [operation] function that takes an element and current accumulator value, and calculates the
next accumulator value.\n

```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
ULongArray.foldRight(initial: R, operation: (ULong, acc: R) -> R): R {\n  var index = lastIndex\n  var
accumulator = initial\n  while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n
return accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from right
to left\n * to each element and current accumulator value.\n * \n * Returns the specified [initial] value if the array is
empty.\n * \n * @param [operation] function that takes an element and current accumulator value, and calculates the
next accumulator value.\n

```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UByteArray.foldRight(initial: R, operation: (UByte, acc: R) -> R): R {\n  var index = lastIndex\n  var
accumulator = initial\n  while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n
return accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from right
to left\n * to each element and current accumulator value.\n * \n * Returns the specified [initial] value if the array is
empty.\n * \n * @param [operation] function that takes an element and current accumulator value, and calculates the
next accumulator value.\n

```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UShortArray.foldRight(initial: R, operation: (UShort, acc: R) -> R): R {\n  var index = lastIndex\n  var
accumulator = initial\n  while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n
return accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying [operation] from right
to left\n * to each element with its index in the original array and current accumulator value.\n * \n * Returns the
specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes the index of an
element, the element itself\n * and current accumulator value, and calculates the next accumulator value.\n

```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UIntArray.foldRightIndexed(initial: R, operation: (index: Int, UInt, acc: R) -> R): R {\n  var index = lastIndex\n
var accumulator = initial\n  while (index >= 0) {\n    accumulator = operation(index, get(index), accumulator)\n
--index\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with [initial] value and applying
[operation] from right to left\n * to each element with its index in the original array and current accumulator value.\n
\n * Returns the specified [initial] value if the array is empty.\n * \n * @param [operation] function that takes the
index of an element, the element itself\n * and current accumulator value, and calculates the next accumulator
value.\n

```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
<R> ULongArray.foldRightIndexed(initial: R, operation: (index: Int, ULong, acc: R) -> R): R {\n  var index =
lastIndex\n  var accumulator = initial\n  while (index >= 0) {\n    accumulator = operation(index, get(index),
accumulator)\n    --index\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with [initial]
value and applying [operation] from right to left\n * to each element with its index in the original array and current
accumulator value.\n * \n * Returns the specified [initial] value if the array is empty.\n * \n * @param [operation]
function that takes the index of an element, the element itself\n * and current accumulator value, and calculates the
next accumulator value.\n

```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UByteArray.foldRightIndexed(initial: R, operation: (index: Int, UByte, acc: R) -> R): R {\n  var index =
lastIndex\n  var accumulator = initial\n  while (index >= 0) {\n    accumulator = operation(index, get(index),
accumulator)\n    --index\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with [initial]
value and applying [operation] from right to left\n * to each element with its index in the original array and current

```

accumulator value.\n \* \n \* Returns the specified [initial] value if the array is empty.\n \* \n \* @param [operation] function that takes the index of an element, the element itself\n \* and current accumulator value, and calculates the next accumulator value.\n

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UShortArray.foldRightIndexed(initial: R, operation: (index: Int, UShort, acc: R) -> R): R {\n    var index =
lastIndex\n    var accumulator = initial\n    while (index >= 0) {\n        accumulator = operation(index, get(index),
accumulator)\n        --index\n    }\n    return accumulator\n}\n\n/**\n * Performs the given [action] on each
element.\n */\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline
fun UIntArray.forEach(action: (UInt) -> Unit): Unit {\n    for (element in this) action(element)\n}\n\n/**\n *
Performs the given [action] on each element.\n */\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.forEach(action: (ULong) -> Unit): Unit {\n    for (element in this) action(element)\n}\n\n/**\n *
Performs the given [action] on each element.\n */\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.forEach(action: (UByte) -> Unit): Unit {\n    for (element in this) action(element)\n}\n\n/**\n *
Performs the given [action] on each element.\n */\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.forEach(action: (UShort) -> Unit): Unit {\n    for (element in this) action(element)\n}\n\n/**\n *
Performs the given [action] on each element, providing sequential index with the element.\n * @param [action]
function that takes the index of an element and the element itself\n * and performs the action on the element.\n */\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.forEachIndexed(action: (index: Int, UInt) -> Unit): Unit {\n    var index = 0\n    for (item in this)
action(index++, item)\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index with
the element.\n * @param [action] function that takes the index of an element and the element itself\n * and performs
the action on the element.\n */\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.forEachIndexed(action: (index: Int, ULong) -> Unit): Unit {\n    var index = 0\n    for (item in this)
action(index++, item)\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index with
the element.\n * @param [action] function that takes the index of an element and the element itself\n * and performs
the action on the element.\n */\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.forEachIndexed(action: (index: Int, UByte) -> Unit): Unit {\n    var index = 0\n    for (item in this)
action(index++, item)\n}\n\n/**\n * Performs the given [action] on each element, providing sequential index with
the element.\n * @param [action] function that takes the index of an element and the element itself\n * and performs
the action on the element.\n */\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.forEachIndexed(action: (index: Int, UShort) -> Unit): Unit {\n    var index = 0\n    for (item in this)
action(index++, item)\n}\n\n@Deprecated("Use maxOrNull instead.")\nReplaceWith("this.maxOrNull()")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.max():
UInt? {\n    return maxOrNull()\n}\n\n@Deprecated("Use maxOrNull instead.")\nReplaceWith("this.maxOrNull()")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.max():
ULong? {\n    return maxOrNull()\n}\n\n@Deprecated("Use maxOrNull instead.")\nReplaceWith("this.maxOrNull()")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",
hiddenSince = "1.6")\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.max():
UByte? {\n    return maxOrNull()\n}\n\n@Deprecated("Use maxOrNull instead.")\nReplaceWith("this.maxOrNull()")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.5",

```

```

hiddenSince = `1.6`)@SinceKotlin("1.3")@ExperimentalUnsignedTypes\npublic fun UShortArray.max():
UShort? {\n    return maxOrNull()\n}\n\n@Deprecated("Use maxByOrNull instead.\",
ReplaceWith("this.maxByOrNull(selector)")\n@DeprecatedSinceKotlin(warningSince = `1.4`, errorSince =
`1.5`, hiddenSince =
`1.6`)@SinceKotlin("1.3")@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
<R : Comparable<R>> UIntArray.maxBy(selector: (UInt) -> R): UInt? {\n    return
maxByOrNull(selector)\n}\n\n@Deprecated("Use maxByOrNull instead.\",
ReplaceWith("this.maxByOrNull(selector)")\n@DeprecatedSinceKotlin(warningSince = `1.4`, errorSince =
`1.5`, hiddenSince =
`1.6`)@SinceKotlin("1.3")@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
<R : Comparable<R>> ULongArray.maxBy(selector: (ULong) -> R): ULong? {\n    return
maxByOrNull(selector)\n}\n\n@Deprecated("Use maxByOrNull instead.\",
ReplaceWith("this.maxByOrNull(selector)")\n@DeprecatedSinceKotlin(warningSince = `1.4`, errorSince =
`1.5`, hiddenSince =
`1.6`)@SinceKotlin("1.3")@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
<R : Comparable<R>> UByteArray.maxBy(selector: (UByte) -> R): UByte? {\n    return
maxByOrNull(selector)\n}\n\n@Deprecated("Use maxByOrNull instead.\",
ReplaceWith("this.maxByOrNull(selector)")\n@DeprecatedSinceKotlin(warningSince = `1.4`, errorSince =
`1.5`, hiddenSince =
`1.6`)@SinceKotlin("1.3")@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
<R : Comparable<R>> UShortArray.maxBy(selector: (UShort) -> R): UShort? {\n    return
maxByOrNull(selector)\n}\n\n/**\n * Returns the first element yielding the largest value of the given function or
`null` if there are no elements.\n * \n * @sample samples.collections.Collections.Aggregates.maxByOrNull\n
*/\n\n@SinceKotlin("1.4")@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UIntArray.maxByOrNull(selector: (UInt) -> R): UInt? {\n    if (isEmpty()) return null\n    var
maxElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return maxElem\n    var maxVal =
selector(maxElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (maxVal < v)
{\n            maxElem = e\n            maxVal = v\n        }\n    }\n    return maxElem\n}\n\n/**\n * Returns the first
element yielding the largest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.maxByOrNull\n
*/\n\n@SinceKotlin("1.4")@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> ULongArray.maxByOrNull(selector: (ULong) -> R): ULong? {\n    if (isEmpty()) return null\n
var maxElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return maxElem\n    var maxVal =
selector(maxElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (maxVal < v)
{\n            maxElem = e\n            maxVal = v\n        }\n    }\n    return maxElem\n}\n\n/**\n * Returns the first
element yielding the largest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.maxByOrNull\n
*/\n\n@SinceKotlin("1.4")@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UByteArray.maxByOrNull(selector: (UByte) -> R): UByte? {\n    if (isEmpty()) return null\n
var maxElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return maxElem\n    var maxVal =
selector(maxElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (maxVal < v)
{\n            maxElem = e\n            maxVal = v\n        }\n    }\n    return maxElem\n}\n\n/**\n * Returns the first
element yielding the largest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.maxByOrNull\n
*/\n\n@SinceKotlin("1.4")@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UShortArray.maxByOrNull(selector: (UShort) -> R): UShort? {\n    if (isEmpty()) return null\n
var maxElem = this[0]\n    val lastIndex = this.lastIndex\n    if (lastIndex == 0) return maxElem\n    var maxVal =
selector(maxElem)\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        val v = selector(e)\n        if (maxVal < v)

```

```

    { \n      maxElem = e \n      maxValue = v \n    } \n } \n return maxElem \n } \n } \n * Returns the largest
value among all values produced by [selector] function \n * applied to each element in the array. \n * \n * If any of
values produced by [selector] function is `NaN`, the returned result is `NaN`. \n * \n * @throws
NoSuchElementException if the array is empty. \n
* \n @SinceKotlin("1.4") \n @OptIn(kotlin.experimental.ExperimentalTypeInference::class) \n @OverloadResolution
ByLambdaReturnType \n @ExperimentalUnsignedTypes \n @kotlin.internal.InlineOnly \n public inline fun
UIntArray.maxOf(selector: (UInt) -> Double): Double { \n   if (isEmpty()) throw NoSuchElementException() \n
var maxValue = selector(this[0]) \n   for (i in 1..lastIndex) { \n     val v = selector(this[i]) \n     maxValue =
maxOf(maxValue, v) \n   } \n   return maxValue \n } \n } \n * Returns the largest value among all values produced
by [selector] function \n * applied to each element in the array. \n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`. \n * \n * @throws NoSuchElementException if the array is empty. \n
* \n @SinceKotlin("1.4") \n @OptIn(kotlin.experimental.ExperimentalTypeInference::class) \n @OverloadResolution
ByLambdaReturnType \n @ExperimentalUnsignedTypes \n @kotlin.internal.InlineOnly \n public inline fun
UIntArray.maxOf(selector: (ULong) -> Double): Double { \n   if (isEmpty()) throw NoSuchElementException() \n
var maxValue = selector(this[0]) \n   for (i in 1..lastIndex) { \n     val v = selector(this[i]) \n     maxValue =
maxOf(maxValue, v) \n   } \n   return maxValue \n } \n } \n * Returns the largest value among all values produced
by [selector] function \n * applied to each element in the array. \n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`. \n * \n * @throws NoSuchElementException if the array is empty. \n
* \n @SinceKotlin("1.4") \n @OptIn(kotlin.experimental.ExperimentalTypeInference::class) \n @OverloadResolution
ByLambdaReturnType \n @ExperimentalUnsignedTypes \n @kotlin.internal.InlineOnly \n public inline fun
UByteArray.maxOf(selector: (UByte) -> Double): Double { \n   if (isEmpty()) throw NoSuchElementException() \n
var maxValue = selector(this[0]) \n   for (i in 1..lastIndex) { \n     val v = selector(this[i]) \n     maxValue =
maxOf(maxValue, v) \n   } \n   return maxValue \n } \n } \n * Returns the largest value among all values produced
by [selector] function \n * applied to each element in the array. \n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`. \n * \n * @throws NoSuchElementException if the array is empty. \n
* \n @SinceKotlin("1.4") \n @OptIn(kotlin.experimental.ExperimentalTypeInference::class) \n @OverloadResolution
ByLambdaReturnType \n @ExperimentalUnsignedTypes \n @kotlin.internal.InlineOnly \n public inline fun
UShortArray.maxOf(selector: (UShort) -> Double): Double { \n   if (isEmpty()) throw
NoSuchElementException() \n   var maxValue = selector(this[0]) \n   for (i in 1..lastIndex) { \n     val v =
selector(this[i]) \n     maxValue = maxOf(maxValue, v) \n   } \n   return maxValue \n } \n } \n * Returns the
largest value among all values produced by [selector] function \n * applied to each element in the array. \n * \n * If
any of values produced by [selector] function is `NaN`, the returned result is `NaN`. \n * \n * @throws
NoSuchElementException if the array is empty. \n
* \n @SinceKotlin("1.4") \n @OptIn(kotlin.experimental.ExperimentalTypeInference::class) \n @OverloadResolution
ByLambdaReturnType \n @ExperimentalUnsignedTypes \n @kotlin.internal.InlineOnly \n public inline fun
UIntArray.maxOf(selector: (UInt) -> Float): Float { \n   if (isEmpty()) throw NoSuchElementException() \n   var
maxValue = selector(this[0]) \n   for (i in 1..lastIndex) { \n     val v = selector(this[i]) \n     maxValue =
maxOf(maxValue, v) \n   } \n   return maxValue \n } \n } \n * Returns the largest value among all values produced
by [selector] function \n * applied to each element in the array. \n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`. \n * \n * @throws NoSuchElementException if the array is empty. \n
* \n @SinceKotlin("1.4") \n @OptIn(kotlin.experimental.ExperimentalTypeInference::class) \n @OverloadResolution
ByLambdaReturnType \n @ExperimentalUnsignedTypes \n @kotlin.internal.InlineOnly \n public inline fun
UIntArray.maxOf(selector: (ULong) -> Float): Float { \n   if (isEmpty()) throw NoSuchElementException() \n
var maxValue = selector(this[0]) \n   for (i in 1..lastIndex) { \n     val v = selector(this[i]) \n     maxValue =
maxOf(maxValue, v) \n   } \n   return maxValue \n } \n } \n * Returns the largest value among all values produced
by [selector] function \n * applied to each element in the array. \n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`. \n * \n * @throws NoSuchElementException if the array is empty. \n
* \n @SinceKotlin("1.4") \n @OptIn(kotlin.experimental.ExperimentalTypeInference::class) \n @OverloadResolution

```

```

ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.maxOf(selector: (UByte) -> Float): Float {\n  if (isEmpty()) throw NoSuchElementException()\n  var
maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    maxValue =
maxOf(maxValue, v)\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.maxOf(selector: (UShort) -> Float): Float {\n  if (isEmpty()) throw NoSuchElementException()\n
var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    maxValue =
maxOf(maxValue, v)\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * @throws NoSuchElementException if the
array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UIntArray.maxOf(selector: (UInt) -> R): R {\n  if (isEmpty()) throw
NoSuchElementException()\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v =
selector(this[i])\n    if (maxValue < v) {\n      maxValue = v\n    }\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the
array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> ULongArray.maxOf(selector: (ULong) -> R): R {\n  if (isEmpty()) throw
NoSuchElementException()\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v =
selector(this[i])\n    if (maxValue < v) {\n      maxValue = v\n    }\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the
array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UByteArray.maxOf(selector: (UByte) -> R): R {\n  if (isEmpty()) throw
NoSuchElementException()\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v =
selector(this[i])\n    if (maxValue < v) {\n      maxValue = v\n    }\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the
array.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UShortArray.maxOf(selector: (UShort) -> R): R {\n  if (isEmpty()) throw
NoSuchElementException()\n  var maxValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v =
selector(this[i])\n    if (maxValue < v) {\n      maxValue = v\n    }\n  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array
or `null` if there are no elements.\n * \n * If any of values produced by [selector] function is `NaN`, the returned
result is `NaN`.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.maxOfOrNull(selector: (UInt) -> Double): Double? {\n  if (isEmpty()) return null\n  var maxValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    maxValue = maxOf(maxValue, v)\n
  }\n  return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n *
applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector]

```

function is `NaN`, the returned result is `NaN`.\n

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nULongArray.maxOfOrNull(selector: (ULong) -> Double): Double? {\n    if (isEmpty()) return null\n    var\n    maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue =\n        maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced\n * by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of\n * values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nUByteArray.maxOfOrNull(selector: (UByte) -> Double): Double? {\n    if (isEmpty()) return null\n    var maxValue\n    = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue,\n        v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector]\n * function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced\n * by [selector] function is `NaN`, the returned result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nUShortArray.maxOfOrNull(selector: (UShort) -> Double): Double? {\n    if (isEmpty()) return null\n    var\n    maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue =\n        maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced\n * by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of\n * values produced by [selector] function is `NaN`, the returned result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nUIntArray.maxOfOrNull(selector: (UInt) -> Float): Float? {\n    if (isEmpty()) return null\n    var maxValue =\n    selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector]\n * function is `NaN`, the returned result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nULongArray.maxOfOrNull(selector: (ULong) -> Float): Float? {\n    if (isEmpty()) return null\n    var maxValue =\n    selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector]\n * function is `NaN`, the returned result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nUByteArray.maxOfOrNull(selector: (UByte) -> Float): Float? {\n    if (isEmpty()) return null\n    var maxValue =\n    selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector]\n * function is `NaN`, the returned result is `NaN`.\n */
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun\nUShortArray.maxOfOrNull(selector: (UShort) -> Float): Float? {\n    if (isEmpty()) return null\n    var maxValue =\n    selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        maxValue = maxOf(maxValue, v)\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value among all values produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector]\n * function is `NaN`, the returned result is `NaN`.\n */
```

```

applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UIntArray.maxOfOrNull(selector: (UInt) -> R): R? {\n if (isEmpty()) return null\n var
maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if (maxValue < v) {\n
maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n * Returns the largest value among all values
produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> ULongArray.maxOfOrNull(selector: (ULong) -> R): R? {\n if (isEmpty()) return null\n var
maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if (maxValue < v) {\n
maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n * Returns the largest value among all values
produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UByteArray.maxOfOrNull(selector: (UByte) -> R): R? {\n if (isEmpty()) return null\n var
maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if (maxValue < v) {\n
maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n * Returns the largest value among all values
produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UShortArray.maxOfOrNull(selector: (UShort) -> R): R? {\n if (isEmpty()) return null\n var
maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if (maxValue < v) {\n
maxValue = v\n }\n }\n return maxValue\n}\n\n/**\n * Returns the largest value according to the
provided [comparator]\n * among all values produced by [selector] function applied to each element in the array.\n *
*\n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UIntArray.maxOfWith(comparator: Comparator<in R>, selector: (UInt) -> R): R {\n if (isEmpty()) throw
NoSuchElementException()\n var maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v =
selector(this[i])\n if (comparator.compare(maxValue, v) < 0) {\n maxValue = v\n }\n }\n return
maxValue\n}\n\n/**\n * Returns the largest value according to the provided [comparator]\n * among all values
produced by [selector] function applied to each element in the array.\n * \n * @throws NoSuchElementException if
the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
ULongArray.maxOfWith(comparator: Comparator<in R>, selector: (ULong) -> R): R {\n if (isEmpty()) throw
NoSuchElementException()\n var maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v =
selector(this[i])\n if (comparator.compare(maxValue, v) < 0) {\n maxValue = v\n }\n }\n return
maxValue\n}\n\n/**\n * Returns the largest value according to the provided [comparator]\n * among all values
produced by [selector] function applied to each element in the array.\n * \n * @throws NoSuchElementException if
the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UByteArray.maxOfWith(comparator: Comparator<in R>, selector: (UByte) -> R): R {\n if (isEmpty()) throw
NoSuchElementException()\n var maxValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v =
selector(this[i])\n if (comparator.compare(maxValue, v) < 0) {\n maxValue = v\n }\n }\n return
maxValue\n}\n\n/**\n * Returns the largest value according to the provided [comparator]\n * among all values

```

produced by [selector] function applied to each element in the array.\n \* \n \* @throws NoSuchElementException if the array is empty.\n

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nUShortArray.maxOfWith(comparator: Comparator<in R>, selector: (UShort) -> R): R {\n    if (isEmpty()) throw\n    NoSuchElementException()\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v =\n        selector(this[i])\n        if (comparator.compare(maxValue, v) < 0) {\n            maxValue = v\n        }\n    }\n    return\n    maxValue\n}\n\n/**\n * Returns the largest value according to the provided [comparator]\n * among all values\n * produced by [selector] function applied to each element in the array or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nUIntArray.maxOfWithOrNull(comparator: Comparator<in R>, selector: (UInt) -> R): R? {\n    if (isEmpty()) return\n    null\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if\n        (comparator.compare(maxValue, v) < 0) {\n            maxValue = v\n        }\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]\n * function applied to each element in the array or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nULongArray.maxOfWithOrNull(comparator: Comparator<in R>, selector: (ULong) -> R): R? {\n    if (isEmpty())\n    return null\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if\n        (comparator.compare(maxValue, v) < 0) {\n            maxValue = v\n        }\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]\n * function applied to each element in the array or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nUByteArray.maxOfWithOrNull(comparator: Comparator<in R>, selector: (UByte) -> R): R? {\n    if (isEmpty())\n    return null\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if\n        (comparator.compare(maxValue, v) < 0) {\n            maxValue = v\n        }\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest value according to the provided [comparator]\n * among all values produced by [selector]\n * function applied to each element in the array or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution\nByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>\nUShortArray.maxOfWithOrNull(comparator: Comparator<in R>, selector: (UShort) -> R): R? {\n    if (isEmpty())\n    return null\n    var maxValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if\n        (comparator.compare(maxValue, v) < 0) {\n            maxValue = v\n        }\n    }\n    return maxValue\n}\n\n/**\n * Returns the largest element or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.maxOrNull(): UInt? {\n    if\n    (isEmpty()) return null\n    var max = this[0]\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        if (max < e) max\n        = e\n    }\n    return max\n}\n\n/**\n * Returns the largest element or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.maxOrNull(): ULong? {\n    if\n    (isEmpty()) return null\n    var max = this[0]\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        if (max < e) max\n        = e\n    }\n    return max\n}\n\n/**\n * Returns the largest element or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.maxOrNull(): UByte? {\n    if\n    (isEmpty()) return null\n    var max = this[0]\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        if (max < e) max\n        = e\n    }\n    return max\n}\n\n/**\n * Returns the largest element or `null` if there are no elements.\n
```

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.maxOrNull(): UShort? {\n    if\n    (isEmpty()) return null\n    var max = this[0]\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        if (max < e) max\n        = e\n    }\n    return max\n}\n\n@Deprecated("Use maxWithOrNull instead.",
```

```

ReplaceWith("this.maxWithOrNull(comparator)")\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince
= \"1.5\", hiddenSince = \"1.6\")\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\npublic fun
UIntArray.maxWith(comparator: Comparator<in UInt>): UInt? {\n    return
maxWithOrNull(comparator)\n}\n\n@Deprecated(\"Use maxWithOrNull instead.\",
ReplaceWith("this.maxWithOrNull(comparator)")\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince
= \"1.5\", hiddenSince = \"1.6\")\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\npublic fun
ULongArray.maxWith(comparator: Comparator<in ULong>): ULong? {\n    return
maxWithOrNull(comparator)\n}\n\n@Deprecated(\"Use maxWithOrNull instead.\",
ReplaceWith("this.maxWithOrNull(comparator)")\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince
= \"1.5\", hiddenSince = \"1.6\")\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\npublic fun
UByteArray.maxWith(comparator: Comparator<in UByte>): UByte? {\n    return
maxWithOrNull(comparator)\n}\n\n@Deprecated(\"Use maxWithOrNull instead.\",
ReplaceWith("this.maxWithOrNull(comparator)")\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince
= \"1.5\", hiddenSince = \"1.6\")\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\npublic fun
UShortArray.maxWith(comparator: Comparator<in UShort>): UShort? {\n    return
maxWithOrNull(comparator)\n}\n\n/**\n * Returns the first element having the largest value according to the
provided [comparator] or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.maxWithOrNull(comparator:
Comparator<in UInt>): UInt? {\n    if (isEmpty()) return null\n    var max = this[0]\n    for (i in 1..lastIndex) {\n
val e = this[i]\n        if (comparator.compare(max, e) < 0) max = e\n    }\n    return max\n}\n\n/**\n * Returns the
first element having the largest value according to the provided [comparator] or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.maxWithOrNull(comparator:
Comparator<in ULong>): ULong? {\n    if (isEmpty()) return null\n    var max = this[0]\n    for (i in 1..lastIndex) {\n
val e = this[i]\n        if (comparator.compare(max, e) < 0) max = e\n    }\n    return max\n}\n\n/**\n * Returns the
first element having the largest value according to the provided [comparator] or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.maxWithOrNull(comparator:
Comparator<in UByte>): UByte? {\n    if (isEmpty()) return null\n    var max = this[0]\n    for (i in 1..lastIndex) {\n
val e = this[i]\n        if (comparator.compare(max, e) < 0) max = e\n    }\n    return max\n}\n\n/**\n * Returns the
first element having the largest value according to the provided [comparator] or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.maxWithOrNull(comparator:
Comparator<in UShort>): UShort? {\n    if (isEmpty()) return null\n    var max = this[0]\n    for (i in 1..lastIndex) {\n
val e = this[i]\n        if (comparator.compare(max, e) < 0) max = e\n    }\n    return
max\n}\n\n@Deprecated(\"Use minOrNull instead.\",
ReplaceWith("this.minOrNull()")\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince = \"1.5\",
hiddenSince = \"1.6\")\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.min(): UInt?
{\n    return minOrNull()\n}\n\n@Deprecated(\"Use minOrNull instead.\",
ReplaceWith("this.minOrNull()")\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince = \"1.5\",
hiddenSince = \"1.6\")\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.min():
ULong? {\n    return minOrNull()\n}\n\n@Deprecated(\"Use minOrNull instead.\",
ReplaceWith("this.minOrNull()")\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince = \"1.5\",
hiddenSince = \"1.6\")\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.min():
UByte? {\n    return minOrNull()\n}\n\n@Deprecated(\"Use minOrNull instead.\",
ReplaceWith("this.minOrNull()")\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince = \"1.5\",
hiddenSince = \"1.6\")\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.min():
UShort? {\n    return minOrNull()\n}\n\n@Deprecated(\"Use minByOrNull instead.\",
ReplaceWith("this.minByOrNull(selector)")\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince =
\"1.5\", hiddenSince =
\"1.6\")\n@SinceKotlin(\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun

```

```

<R : Comparable<R>> UIntArray.minBy(selector: (UInt) -> R): UInt? {\n  return
minByOrNull(selector)\n}\n\n@Deprecated("Use minByOrNull instead.\",
ReplaceWith("this.minByOrNull(selector)\")\n)\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince =
"1.6")\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
<R : Comparable<R>> ULongArray.minBy(selector: (ULong) -> R): ULong? {\n  return
minByOrNull(selector)\n}\n\n@Deprecated("Use minByOrNull instead.\",
ReplaceWith("this.minByOrNull(selector)\")\n)\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince =
"1.6")\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
<R : Comparable<R>> UByteArray.minBy(selector: (UByte) -> R): UByte? {\n  return
minByOrNull(selector)\n}\n\n@Deprecated("Use minByOrNull instead.\",
ReplaceWith("this.minByOrNull(selector)\")\n)\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince =
"1.5", hiddenSince =
"1.6")\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
<R : Comparable<R>> UShortArray.minBy(selector: (UShort) -> R): UShort? {\n  return
minByOrNull(selector)\n}\n\n**\n * Returns the first element yielding the smallest value of the given function or
`null` if there are no elements.\n * \n * @sample samples.collections.Collections.Aggregates.minByOrNull\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UIntArray.minByOrNull(selector: (UInt) -> R): UInt? {\n  if (isEmpty()) return null\n  var
minElem = this[0]\n  val lastIndex = this.lastIndex\n  if (lastIndex == 0) return minElem\n  var minValue =
selector(minElem)\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    val v = selector(e)\n    if (minValue > v)
{\n      minElem = e\n      minValue = v\n    }\n  }\n  return minElem\n}\n\n**\n * Returns the first
element yielding the smallest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.minByOrNull\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> ULongArray.minByOrNull(selector: (ULong) -> R): ULong? {\n  if (isEmpty()) return null\n
var minElem = this[0]\n  val lastIndex = this.lastIndex\n  if (lastIndex == 0) return minElem\n  var minValue =
selector(minElem)\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    val v = selector(e)\n    if (minValue > v)
{\n      minElem = e\n      minValue = v\n    }\n  }\n  return minElem\n}\n\n**\n * Returns the first
element yielding the smallest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.minByOrNull\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UByteArray.minByOrNull(selector: (UByte) -> R): UByte? {\n  if (isEmpty()) return null\n
var minElem = this[0]\n  val lastIndex = this.lastIndex\n  if (lastIndex == 0) return minElem\n  var minValue =
selector(minElem)\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    val v = selector(e)\n    if (minValue > v)
{\n      minElem = e\n      minValue = v\n    }\n  }\n  return minElem\n}\n\n**\n * Returns the first
element yielding the smallest value of the given function or `null` if there are no elements.\n * \n * @sample
samples.collections.Collections.Aggregates.minByOrNull\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UShortArray.minByOrNull(selector: (UShort) -> R): UShort? {\n  if (isEmpty()) return null\n
var minElem = this[0]\n  val lastIndex = this.lastIndex\n  if (lastIndex == 0) return minElem\n  var minValue =
selector(minElem)\n  for (i in 1..lastIndex) {\n    val e = this[i]\n    val v = selector(e)\n    if (minValue > v)
{\n      minElem = e\n      minValue = v\n    }\n  }\n  return minElem\n}\n\n**\n * Returns the smallest
value among all values produced by [selector] function\n * applied to each element in the array.\n * \n * If any of
values produced by [selector] function is `NaN`, the returned result is `NaN`.\n * \n * @throws
NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution

```

```

ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.minOf(selector: (UInt) -> Double): Double {\n  if (isEmpty()) throw NoSuchElementException()\n
var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue =
minOf(minValue, v)\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.minOf(selector: (ULong) -> Double): Double {\n  if (isEmpty()) throw NoSuchElementException()\n
var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue =
minOf(minValue, v)\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.minOf(selector: (UByte) -> Double): Double {\n  if (isEmpty()) throw NoSuchElementException()\n
var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue =
minOf(minValue, v)\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.minOf(selector: (UShort) -> Double): Double {\n  if (isEmpty()) throw NoSuchElementException()\n
var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue =
minOf(minValue, v)\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.minOf(selector: (UInt) -> Float): Float {\n  if (isEmpty()) throw NoSuchElementException()\n  var
minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue =
minOf(minValue, v)\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.minOf(selector: (ULong) -> Float): Float {\n  if (isEmpty()) throw NoSuchElementException()\n
var minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue =
minOf(minValue, v)\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.minOf(selector: (UByte) -> Float): Float {\n  if (isEmpty()) throw NoSuchElementException()\n  var
minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue =
minOf(minValue, v)\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n * \n * @throws NoSuchElementException if the array is empty.\n
*/\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.minOf(selector: (UShort) -> Float): Float {\n    if (isEmpty()) throw NoSuchElementException()\n
var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue =
minOf(minValue, v)\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced
by [selector] function\n * applied to each element in the array.\n * @throws NoSuchElementException if the
array is empty.\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UIntArray.minOf(selector: (UInt) -> R): R {\n    if (isEmpty()) throw
NoSuchElementException()\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v =
selector(this[i])\n        if (minValue > v) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the
array.\n * @throws NoSuchElementException if the array is empty.\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> ULongArray.minOf(selector: (ULong) -> R): R {\n    if (isEmpty()) throw
NoSuchElementException()\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v =
selector(this[i])\n        if (minValue > v) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the
array.\n * @throws NoSuchElementException if the array is empty.\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UByteArray.minOf(selector: (UByte) -> R): R {\n    if (isEmpty()) throw
NoSuchElementException()\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v =
selector(this[i])\n        if (minValue > v) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the
array.\n * @throws NoSuchElementException if the array is empty.\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UShortArray.minOfOrNull(selector: (UShort) -> R): R {\n    if (isEmpty()) throw
NoSuchElementException()\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v =
selector(this[i])\n        if (minValue > v) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n * applied to each element in the array
or `null` if there are no elements.\n * If any of values produced by [selector] function is `NaN`, the returned
result is `NaN`.\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.minOfOrNull(selector: (UInt) -> Double): Double? {\n    if (isEmpty()) return null\n    var minValue =
selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n
    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n *
applied to each element in the array or `null` if there are no elements.\n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n

```

```

*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.minOfOrNull(selector: (ULong) -> Double): Double? {\n    if (isEmpty()) return null\n    var minValue
= selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        minValue = minOf(minValue, v)\n
    }\n    return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n

```

```

* applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by
[selector] function is `NaN`, the returned result is `NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.minOrNull(selector: (UByte) -> Double): Double? {\n  if (isEmpty()) return null\n  var minValue
= selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n
  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n
* applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by
[selector] function is `NaN`, the returned result is `NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.minOrNull(selector: (UShort) -> Double): Double? {\n  if (isEmpty()) return null\n  var minValue
= selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n
  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n
* applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by
[selector] function is `NaN`, the returned result is `NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.minOrNull(selector: (UInt) -> Float): Float? {\n  if (isEmpty()) return null\n  var minValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n
  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n
* applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.minOrNull(selector: (ULong) -> Float): Float? {\n  if (isEmpty()) return null\n  var minValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n
  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n
* applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.minOrNull(selector: (UByte) -> Float): Float? {\n  if (isEmpty()) return null\n  var minValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n
  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n
* applied to each element in the array or `null` if there are no elements.\n * \n * If any of values produced by [selector]
function is `NaN`, the returned result is `NaN`.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.minOrNull(selector: (UShort) -> Float): Float? {\n  if (isEmpty()) return null\n  var minValue =
selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    minValue = minOf(minValue, v)\n
  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values produced by [selector] function\n
* applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UIntArray.minOrNull(selector: (UInt) -> R): R? {\n  if (isEmpty()) return null\n  var
minValue = selector(this[0])\n  for (i in 1..lastIndex) {\n    val v = selector(this[i])\n    if (minValue > v) {\n
      minValue = v\n    }\n  }\n  return minValue\n}\n\n/**\n * Returns the smallest value among all values

```

```

produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> ULongArray.minOfOrNull(selector: (ULong) -> R): R? {\n if (isEmpty()) return null\n var
minValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if (minValue > v) {\n
minValue = v\n }\n }\n return minValue}\n\n/**\n * Returns the smallest value among all values
produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UByteArray.minOfOrNull(selector: (UByte) -> R): R? {\n if (isEmpty()) return null\n var
minValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if (minValue > v) {\n
minValue = v\n }\n }\n return minValue}\n\n/**\n * Returns the smallest value among all values
produced by [selector] function\n * applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R :
Comparable<R>> UShortArray.minOfOrNull(selector: (UShort) -> R): R? {\n if (isEmpty()) return null\n var
minValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v = selector(this[i])\n if (minValue > v) {\n
minValue = v\n }\n }\n return minValue}\n\n/**\n * Returns the smallest value according to the
provided [comparator]\n * among all values produced by [selector] function applied to each element in the array.\n *
\n * @throws NoSuchElementException if the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UIntArray.minOfWith(comparator: Comparator<in R>, selector: (UInt) -> R): R {\n if (isEmpty()) throw
NoSuchElementException()\n var minValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v =
selector(this[i])\n if (comparator.compare(minValue, v) > 0) {\n minValue = v\n }\n }\n return
minValue}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values
produced by [selector] function applied to each element in the array.\n * \n * @throws NoSuchElementException if
the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
ULongArray.minOfWith(comparator: Comparator<in R>, selector: (ULong) -> R): R {\n if (isEmpty()) throw
NoSuchElementException()\n var minValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v =
selector(this[i])\n if (comparator.compare(minValue, v) > 0) {\n minValue = v\n }\n }\n return
minValue}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values
produced by [selector] function applied to each element in the array.\n * \n * @throws NoSuchElementException if
the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UByteArray.minOfWith(comparator: Comparator<in R>, selector: (UByte) -> R): R {\n if (isEmpty()) throw
NoSuchElementException()\n var minValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v =
selector(this[i])\n if (comparator.compare(minValue, v) > 0) {\n minValue = v\n }\n }\n return
minValue}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values
produced by [selector] function applied to each element in the array.\n * \n * @throws NoSuchElementException if
the array is empty.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UShortArray.minOfWith(comparator: Comparator<in R>, selector: (UShort) -> R): R {\n if (isEmpty()) throw
NoSuchElementException()\n var minValue = selector(this[0])\n for (i in 1..lastIndex) {\n val v =

```

```

selector(this[i])\n    if (comparator.compare(minValue, v) > 0) {\n        minValue = v\n    }\n }\n return
minValue\n}\n\n/**\n * Returns the smallest value according to the provided [comparator]\n * among all values
produced by [selector] function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UIntArray.minOfWithOrNull(comparator: Comparator<in R>, selector: (UInt) -> R): R? {\n    if (isEmpty()) return
null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if
(comparator.compare(minValue, v) > 0) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n *
Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
ULongArray.minOfWithOrNull(comparator: Comparator<in R>, selector: (ULong) -> R): R? {\n    if (isEmpty())
return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if
(comparator.compare(minValue, v) > 0) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n *
Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UByteArray.minOfWithOrNull(comparator: Comparator<in R>, selector: (UByte) -> R): R? {\n    if (isEmpty())
return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if
(comparator.compare(minValue, v) > 0) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n *
Returns the smallest value according to the provided [comparator]\n * among all values produced by [selector]
function applied to each element in the array or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UShortArray.minOfWithOrNull(comparator: Comparator<in R>, selector: (UShort) -> R): R? {\n    if (isEmpty())
return null\n    var minValue = selector(this[0])\n    for (i in 1..lastIndex) {\n        val v = selector(this[i])\n        if
(comparator.compare(minValue, v) > 0) {\n            minValue = v\n        }\n    }\n    return minValue\n}\n\n/**\n *
Returns the smallest element or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.minOrNull(): UInt? {\n    if
(isEmpty()) return null\n    var min = this[0]\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        if (min > e) min =
e\n    }\n    return min\n}\n\n/**\n * Returns the smallest element or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.minOrNull(): ULong? {\n    if
(isEmpty()) return null\n    var min = this[0]\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        if (min > e) min =
e\n    }\n    return min\n}\n\n/**\n * Returns the smallest element or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.minOrNull(): UByte? {\n    if
(isEmpty()) return null\n    var min = this[0]\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        if (min > e) min =
e\n    }\n    return min\n}\n\n/**\n * Returns the smallest element or `null` if there are no elements.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.minOrNull(): UShort? {\n    if
(isEmpty()) return null\n    var min = this[0]\n    for (i in 1..lastIndex) {\n        val e = this[i]\n        if (min > e) min =
e\n    }\n    return min\n}\n\n@Deprecated("Use minWithOrNull instead.")\nReplaceWith("this.minWithOrNull(comparator)")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince
= "1.5", hiddenSince = "1.6")\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun
UIntArray.minWith(comparator: Comparator<in UInt>): UInt? {\n    return
minWithOrNull(comparator)\n}\n\n@Deprecated("Use minWithOrNull instead.")\nReplaceWith("this.minWithOrNull(comparator)")\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince
= "1.5", hiddenSince = "1.6")\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun

```

```

ULongArray.minWith(comparator: Comparator<in ULong>): ULong? {\n  return
minWithOrNull(comparator)\n}\n\n@Deprecated(\\"Use minWithOrNull instead.\",
ReplaceWith(\\"this.minWithOrNull(comparator)\")\n)\n@DeprecatedSinceKotlin(warningSince = \\"1.4\", errorSince
= \\"1.5\", hiddenSince = \\"1.6\")\n@SinceKotlin(\\"1.3\")\n@ExperimentalUnsignedTypes\npublic fun
UByteArray.minWith(comparator: Comparator<in UByte>): UByte? {\n  return
minWithOrNull(comparator)\n}\n\n@Deprecated(\\"Use minWithOrNull instead.\",
ReplaceWith(\\"this.minWithOrNull(comparator)\")\n)\n@DeprecatedSinceKotlin(warningSince = \\"1.4\", errorSince
= \\"1.5\", hiddenSince = \\"1.6\")\n@SinceKotlin(\\"1.3\")\n@ExperimentalUnsignedTypes\npublic fun
UShortArray.minWith(comparator: Comparator<in UShort>): UShort? {\n  return
minWithOrNull(comparator)\n}\n\n/**\n * Returns the first element having the smallest value according to the
provided [comparator] or `null` if there are no elements.\n
*\n*\n@SinceKotlin(\\"1.4\")\n@ExperimentalUnsignedTypes\npublic fun UIntArray.minWithOrNull(comparator:
Comparator<in UInt>): UInt? {\n  if (isEmpty()) return null\n  var min = this[0]\n  for (i in 1..lastIndex) {\n
val e = this[i]\n    if (comparator.compare(min, e) > 0) min = e\n  }\n  return min\n}\n\n/**\n * Returns the first
element having the smallest value according to the provided [comparator] or `null` if there are no elements.\n
*\n*\n@SinceKotlin(\\"1.4\")\n@ExperimentalUnsignedTypes\npublic fun ULongArray.minWithOrNull(comparator:
Comparator<in ULong>): ULong? {\n  if (isEmpty()) return null\n  var min = this[0]\n  for (i in 1..lastIndex) {\n
val e = this[i]\n    if (comparator.compare(min, e) > 0) min = e\n  }\n  return min\n}\n\n/**\n * Returns the
first element having the smallest value according to the provided [comparator] or `null` if there are no elements.\n
*\n*\n@SinceKotlin(\\"1.4\")\n@ExperimentalUnsignedTypes\npublic fun UByteArray.minWithOrNull(comparator:
Comparator<in UByte>): UByte? {\n  if (isEmpty()) return null\n  var min = this[0]\n  for (i in 1..lastIndex) {\n
val e = this[i]\n    if (comparator.compare(min, e) > 0) min = e\n  }\n  return min\n}\n\n/**\n * Returns the
first element having the smallest value according to the provided [comparator] or `null` if there are no elements.\n
*\n*\n@SinceKotlin(\\"1.4\")\n@ExperimentalUnsignedTypes\npublic fun UShortArray.minWithOrNull(comparator:
Comparator<in UShort>): UShort? {\n  if (isEmpty()) return null\n  var min = this[0]\n  for (i in 1..lastIndex) {\n
val e = this[i]\n    if (comparator.compare(min, e) > 0) min = e\n  }\n  return min\n}\n\n/**\n * Returns
`true` if the array has no elements.\n * \n * @sample samples.collections.Collections.Aggregates.none\n
*\n*\n@SinceKotlin(\\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.none(): Boolean {\n  return isEmpty()\n}\n\n/**\n * Returns `true` if the array has no elements.\n * \n *
@sample samples.collections.Collections.Aggregates.none\n
*\n*\n@SinceKotlin(\\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.none(): Boolean {\n  return isEmpty()\n}\n\n/**\n * Returns `true` if the array has no elements.\n * \n *
@sample samples.collections.Collections.Aggregates.none\n
*\n*\n@SinceKotlin(\\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.none(): Boolean {\n  return isEmpty()\n}\n\n/**\n * Returns `true` if the array has no elements.\n * \n *
@sample samples.collections.Collections.Aggregates.none\n
*\n*\n@SinceKotlin(\\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.none(): Boolean {\n  return isEmpty()\n}\n\n/**\n * Returns `true` if no elements match the given
[predicate].\n * \n * @sample samples.collections.Collections.Aggregates.noneWithPredicate\n
*\n*\n@SinceKotlin(\\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.none(predicate: (UInt) -> Boolean): Boolean {\n  for (element in this) if (predicate(element)) return
false\n  return true\n}\n\n/**\n * Returns `true` if no elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n
*\n*\n@SinceKotlin(\\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.none(predicate: (ULong) -> Boolean): Boolean {\n  for (element in this) if (predicate(element)) return
false\n  return true\n}\n\n/**\n * Returns `true` if no elements match the given [predicate].\n * \n * @sample
samples.collections.Collections.Aggregates.noneWithPredicate\n
*\n*\n@SinceKotlin(\\"1.3\")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun

```

```

UByteArray.none(predicate: (UByte) -> Boolean): Boolean {
    for (element in this) if (predicate(element)) return false
    return true
}

Returns `true` if no elements match the given [predicate].

@sample
samples.collections.Collections.Aggregates.noneWithPredicate

@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UShortArray.none(predicate: (UShort) -> Boolean): Boolean {
    for (element in this) if (predicate(element)) return false
    return true
}

Performs the given [action] on each element and returns the array itself afterwards.

@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UIntArray.forEach(action: (UInt) -> Unit): UIntArray {
    return apply { for (element in this) action(element) }
}

Performs the given [action] on each element and returns the array itself afterwards.

@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
ULongArray.forEach(action: (ULong) -> Unit): ULongArray {
    return apply { for (element in this) action(element) }
}

Performs the given [action] on each element and returns the array itself afterwards.

@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UByteArray.forEach(action: (UByte) -> Unit): UByteArray {
    return apply { for (element in this) action(element) }
}

Performs the given [action] on each element and returns the array itself afterwards.

@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UShortArray.forEach(action: (UShort) -> Unit): UShortArray {
    return apply { for (element in this) action(element) }
}

Performs the given [action] on each element, providing sequential index with the element,
and returns the array itself afterwards.

@param [action] function that takes the index of an element and the element itself
and performs the action on the element.

@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UIntArray.forEachIndexed(action: (index: Int, UInt) -> Unit): UIntArray {
    return apply {
        forEachIndexed(action)
    }
}

Performs the given [action] on each element, providing sequential index with the element,
and returns the array itself afterwards.

@param [action] function that takes the index of an element and the element itself
and performs the action on the element.

@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
ULongArray.forEachIndexed(action: (index: Int, ULong) -> Unit): ULongArray {
    return apply {
        forEachIndexed(action)
    }
}

Performs the given [action] on each element, providing sequential index with the element,
and returns the array itself afterwards.

@param [action] function that takes the index of an element and the element itself
and performs the action on the element.

@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UByteArray.forEachIndexed(action: (index: Int, UByte) -> Unit): UByteArray {
    return apply {
        forEachIndexed(action)
    }
}

Performs the given [action] on each element, providing sequential index with the element,
and returns the array itself afterwards.

@param [action] function that takes the index of an element and the element itself
and performs the action on the element.

@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UShortArray.forEachIndexed(action: (index: Int, UShort) -> Unit): UShortArray {
    return apply {
        forEachIndexed(action)
    }
}

Accumulates value starting with the first element and applying [operation] from left to right
to current accumulator value and each element.

Throws an exception if this array is empty. If the array can be empty in an expected way,
please use [reduceOrNull] instead. It returns `null` when receiver is empty.

@param [operation] function that takes current accumulator value and an element,
and calculates the next accumulator value.

@sample
samples.collections.Collections.Aggregates.reduce

@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UIntArray.reduce(operation: (acc: UInt, UInt) -> UInt): UInt {
    if (isEmpty()) throw
        UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator = operation(accumulator, this[index])
    }
    return accumulator
}

Accumulates value starting with the first element and applying [operation] from left to right
to current accumulator value and each element.

Throws an exception if this array is empty. If the array can be empty

```

in an expected way, please use [reduceOrNull] instead. It returns `null` when its receiver is empty.

`@param [operation]` function that takes current accumulator value and an element and calculates the next accumulator value.

`@sample samples.collections.Collections.Aggregates.reduce`

```
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.reduce(operation: (acc: ULong, ULong) -> ULong): ULong {\n if (isEmpty())\n throw
UnsupportedOperationException("Empty array can't be reduced.")\n var accumulator = this[0]\n for (index in
1..lastIndex) {\n accumulator = operation(accumulator, this[index])\n }\n return accumulator\n}\n\n/**\n Accumulates value starting with the first element and applying [operation] from left to right\n * to current
accumulator value and each element.\n * \n * Throws an exception if this array is empty. If the array can be empty
in an expected way, please use [reduceOrNull] instead. It returns `null` when its receiver is empty.\n * \n *
@param [operation] function that takes current accumulator value and an element,\n * and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduce\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.reduce(operation: (acc: UByte, UByte) -> UByte): UByte {\n if (isEmpty())\n throw
UnsupportedOperationException("Empty array can't be reduced.")\n var accumulator = this[0]\n for (index in
1..lastIndex) {\n accumulator = operation(accumulator, this[index])\n }\n return accumulator\n}\n\n/**\n Accumulates value starting with the first element and applying [operation] from left to right\n * to current
accumulator value and each element.\n * \n * Throws an exception if this array is empty. If the array can be empty
in an expected way, please use [reduceOrNull] instead. It returns `null` when its receiver is empty.\n * \n *
@param [operation] function that takes current accumulator value and an element,\n * and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduce\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.reduce(operation: (acc: UShort, UShort) -> UShort): UShort {\n if (isEmpty())\n throw
UnsupportedOperationException("Empty array can't be reduced.")\n var accumulator = this[0]\n for (index in
1..lastIndex) {\n accumulator = operation(accumulator, this[index])\n }\n return accumulator\n}\n\n/**\n Accumulates value starting with the first element and applying [operation] from left to right\n * to current
accumulator value and each element with its index in the original array.\n * \n * Throws an exception if this array is
empty. If the array can be empty in an expected way, please use [reduceIndexedOrNull] instead. It returns `null`
when its receiver is empty.\n * \n * @param [operation] function that takes the index of an element, current
accumulator value and the element itself,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduce\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.reduceIndexed(operation: (index: Int, acc: UInt, UInt) -> UInt): UInt {\n if (isEmpty())\n throw
UnsupportedOperationException("Empty array can't be reduced.")\n var accumulator = this[0]\n for (index in
1..lastIndex) {\n accumulator = operation(index, accumulator, this[index])\n }\n return
accumulator\n}\n\n/**\n Accumulates value starting with the first element and applying [operation] from left to
right\n * to current accumulator value and each element with its index in the original array.\n * \n * Throws an
exception if this array is empty. If the array can be empty in an expected way, please use [reduceIndexedOrNull]
instead. It returns `null` when its receiver is empty.\n * \n * @param [operation] function that takes the index of an
element, current accumulator value and the element itself,\n * and calculates the next accumulator value.\n * \n *
@sample samples.collections.Collections.Aggregates.reduce\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.reduceIndexed(operation: (index: Int, acc: ULong, ULong) -> ULong): ULong {\n if (isEmpty())\n
throw UnsupportedOperationException("Empty array can't be reduced.")\n var accumulator = this[0]\n for
(index in 1..lastIndex) {\n accumulator = operation(index, accumulator, this[index])\n }\n return
accumulator\n}\n\n/**\n Accumulates value starting with the first element and applying [operation] from left to
right\n * to current accumulator value and each element with its index in the original array.\n * \n * Throws an
exception if this array is empty. If the array can be empty in an expected way, please use [reduceIndexedOrNull]
```

instead. It returns `null` when its receiver is empty.

```

    @param [operation] function that takes the index of an
    element, current accumulator value and the element itself,
    and calculates the next accumulator value.
    @sample samples.collections.Collections.Aggregates.reduce
*/
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UByteArray.reduceIndexed(operation: (index: Int, acc: UByte, UByte) -> UByte): UByte {
    if (isEmpty())
    throw UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator = operation(index, accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first element and applying [operation] from left to right to current accumulator value and each element with its index in the original array. Throws an exception if this array is empty. If the array can be empty in an expected way, please use [reduceIndexedOrNull] instead. It returns `null` when its receiver is empty.

```

    @param [operation] function that takes the index of an
    element, current accumulator value and the element itself,
    and calculates the next accumulator value.
    @sample samples.collections.Collections.Aggregates.reduce
*/
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UShortArray.reduceIndexed(operation: (index: Int, acc: UShort, UShort) -> UShort): UShort {
    if (isEmpty())
    throw UnsupportedOperationException("Empty array can't be reduced.")
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator = operation(index, accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first element and applying [operation] from left to right to current accumulator value and each element with its index in the original array. Returns `null` if the array is empty.

```

    @param [operation] function that takes the index of an element, current accumulator
    value and the element itself,
    and calculates the next accumulator value.
    @sample
    samples.collections.Collections.Aggregates.reduceOrNull
*/
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UIntArray.reduceIndexedOrNull(operation: (index: Int, acc: UInt, UInt) -> UInt): UInt? {
    if (isEmpty())
    return null
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator = operation(index, accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first element and applying [operation] from left to right to current accumulator value and each element with its index in the original array. Returns `null` if the array is empty.

```

    @param [operation] function that takes the
    index of an element, current accumulator value and the element itself,
    and calculates the next accumulator
    value.
    @sample samples.collections.Collections.Aggregates.reduceOrNull
*/
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
ULongArray.reduceIndexedOrNull(operation: (index: Int, acc: ULong, ULong) -> ULong): ULong? {
    if (isEmpty())
    return null
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator =
        operation(index, accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first element and applying [operation] from left to right to current accumulator value and each element with its index in the original array. Returns `null` if the array is empty.

```

    @param [operation]
    function that takes the index of an element, current accumulator value and the element itself,
    and calculates the
    next accumulator value.
    @sample samples.collections.Collections.Aggregates.reduceOrNull
*/
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UByteArray.reduceIndexedOrNull(operation: (index: Int, acc: UByte, UByte) -> UByte): UByte? {
    if (isEmpty())
    return null
    var accumulator = this[0]
    for (index in 1..lastIndex) {
        accumulator =
        operation(index, accumulator, this[index])
    }
    return accumulator
}

```

Accumulates value starting with the first element and applying [operation] from left to right to current accumulator value and each element with its index in the original array. Returns `null` if the array is empty.

```

    @param [operation]
    function that takes the index of an element, current accumulator value and the element itself,
    and calculates the
    next accumulator value.
    @sample samples.collections.Collections.Aggregates.reduceOrNull
*/
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UShortArray.reduceIndexedOrNull(operation: (index: Int, acc: UShort, UShort) -> UShort): UShort? {
    if

```

```

(isEmpty())\n    return null\n    var accumulator = this[0]\n    for (index in 1..lastIndex) {\n        accumulator =
operation(index, accumulator, this[index])\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value starting
with the first element and applying [operation] from left to right\n * to current accumulator value and each
element.\n * \n * Returns `null` if the array is empty.\n * \n * @param [operation] function that takes current
accumulator value and an element,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceOrNull\n
*/\n\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun UIntArray.reduceOrNull(operation: (acc: UInt, UInt) -> UInt): UInt?
{\n    if (isEmpty())\n        return null\n    var accumulator = this[0]\n    for (index in 1..lastIndex) {\n
accumulator = operation(accumulator, this[index])\n    }\n    return accumulator\n}\n\n/**\n * Accumulates value
starting with the first element and applying [operation] from left to right\n * to current accumulator value and each
element.\n * \n * Returns `null` if the array is empty.\n * \n * @param [operation] function that takes current
accumulator value and an element,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceOrNull\n
*/\n\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun ULongArray.reduceOrNull(operation: (acc: ULong, ULong) ->
ULong): ULong? {\n    if (isEmpty())\n        return null\n    var accumulator = this[0]\n    for (index in 1..lastIndex)
{\n        accumulator = operation(accumulator, this[index])\n    }\n    return accumulator\n}\n\n/**\n * Accumulates
value starting with the first element and applying [operation] from left to right\n * to current accumulator value and
each element.\n * \n * Returns `null` if the array is empty.\n * \n * @param [operation] function that takes current
accumulator value and an element,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceOrNull\n
*/\n\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun UByteArray.reduceOrNull(operation: (acc: UByte, UByte) ->
UByte): UByte? {\n    if (isEmpty())\n        return null\n    var accumulator = this[0]\n    for (index in 1..lastIndex)
{\n        accumulator = operation(accumulator, this[index])\n    }\n    return accumulator\n}\n\n/**\n * Accumulates
value starting with the first element and applying [operation] from left to right\n * to current accumulator value and
each element.\n * \n * Returns `null` if the array is empty.\n * \n * @param [operation] function that takes current
accumulator value and an element,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceOrNull\n
*/\n\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun UShortArray.reduceOrNull(operation: (acc: UShort, UShort) ->
UShort): UShort? {\n    if (isEmpty())\n        return null\n    var accumulator = this[0]\n    for (index in 1..lastIndex)
{\n        accumulator = operation(accumulator, this[index])\n    }\n    return accumulator\n}\n\n/**\n * Accumulates
value starting with the last element and applying [operation] from right to left\n * to each element and current
accumulator value.\n * \n * Throws an exception if this array is empty. If the array can be empty in an expected
way,\n * please use [reduceRightOrNull] instead. It returns `null` when its receiver is empty.\n * \n * @param
[operation] function that takes an element and current accumulator value,\n * and calculates the next accumulator
value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceRight\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.reduceRight(operation: (UInt, acc: UInt) -> UInt): UInt {\n    var index = lastIndex\n    if (index < 0)
throw UnsupportedOperationException("Empty array can't be reduced.")\n    var accumulator = get(index--)\n
while (index >= 0) {\n        accumulator = operation(get(index--), accumulator)\n    }\n    return
accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to
left\n * to each element and current accumulator value.\n * \n * Throws an exception if this array is empty. If the
array can be empty in an expected way,\n * please use [reduceRightOrNull] instead. It returns `null` when its
receiver is empty.\n * \n * @param [operation] function that takes an element and current accumulator value,\n *
and calculates the next accumulator value.\n * \n * @sample

```

```

samples.collections.Collections.Aggregates.reduceRight\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.reduceRight(operation: (ULong, acc: ULong) -> ULong): ULong {\n  var index = lastIndex\n  if
(index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n  var accumulator =
get(index--)\n  while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n  return
accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to
left\n * to each element and current accumulator value.\n * \n * Throws an exception if this array is empty. If the
array can be empty in an expected way,\n * please use [reduceRightOrNull] instead. It returns `null` when its
receiver is empty.\n * \n * @param [operation] function that takes an element and current accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.reduceRight(operation: (UByte, acc: UByte) -> UByte): UByte {\n  var index = lastIndex\n  if (index
< 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n  var accumulator = get(index--)\n
while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n  return
accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to
left\n * to each element and current accumulator value.\n * \n * Throws an exception if this array is empty. If the
array can be empty in an expected way,\n * please use [reduceRightOrNull] instead. It returns `null` when its
receiver is empty.\n * \n * @param [operation] function that takes an element and current accumulator value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.reduceRight(operation: (UShort, acc: UShort) -> UShort): UShort {\n  var index = lastIndex\n  if
(index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n  var accumulator =
get(index--)\n  while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n  return
accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to
left\n * to each element with its index in the original array and current accumulator value.\n * \n * Throws an
exception if this array is empty. If the array can be empty in an expected way,\n * please use
[reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n * @param [operation]
function that takes the index of an element, the element itself and current accumulator value,\n * and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceRight\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.reduceRightIndexed(operation: (index: Int, UInt, acc: UInt) -> UInt): UInt {\n  var index = lastIndex\n
if (index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n  var accumulator =
get(index--)\n  while (index >= 0) {\n    accumulator = operation(index, get(index), accumulator)\n    --index\n
  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation]
from right to left\n * to each element with its index in the original array and current accumulator value.\n * \n *
Throws an exception if this array is empty. If the array can be empty in an expected way,\n * please use
[reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n * @param [operation]
function that takes the index of an element, the element itself and current accumulator value,\n * and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceRight\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.reduceRightIndexed(operation: (index: Int, ULong, acc: ULong) -> ULong): ULong {\n  var index =
lastIndex\n  if (index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n  var
accumulator = get(index--)\n  while (index >= 0) {\n    accumulator = operation(index, get(index),
accumulator)\n    --index\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with the last
element and applying [operation] from right to left\n * to each element with its index in the original array and
current accumulator value.\n * \n * Throws an exception if this array is empty. If the array can be empty in an

```

```

expected way,\n * please use [reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n
 * @param [operation] function that takes the index of an element, the element itself and current accumulator
value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.reduceRightIndexed(operation: (index: Int, UByte, acc: UByte) -> UByte): UByte {\n  var index =
lastIndex\n  if (index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n  var
accumulator = get(index--)\n  while (index >= 0) {\n    accumulator = operation(index, get(index),
accumulator)\n    --index\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with the last
element and applying [operation] from right to left\n * to each element with its index in the original array and
current accumulator value.\n * \n * Throws an exception if this array is empty. If the array can be empty in an
expected way,\n * please use [reduceRightIndexedOrNull] instead. It returns `null` when its receiver is empty.\n * \n
 * @param [operation] function that takes the index of an element, the element itself and current accumulator
value,\n * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRight\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.reduceRightIndexed(operation: (index: Int, UShort, acc: UShort) -> UShort): UShort {\n  var index =
lastIndex\n  if (index < 0) throw UnsupportedOperationException("Empty array can't be reduced.")\n  var
accumulator = get(index--)\n  while (index >= 0) {\n    accumulator = operation(index, get(index),
accumulator)\n    --index\n  }\n  return accumulator\n}\n\n/**\n * Accumulates value starting with the last
element and applying [operation] from right to left\n * to each element with its index in the original array and
current accumulator value.\n * \n * Returns `null` if the array is empty.\n * \n * @param [operation] function that
takes the index of an element, the element itself and current accumulator value,\n * and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.reduceRightIndexedOrNull(operation: (index: Int, UInt, acc: UInt) -> UInt): UInt? {\n  var index =
lastIndex\n  if (index < 0) return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n
accumulator = operation(index, get(index), accumulator)\n    --index\n  }\n  return accumulator\n}\n\n/**\n *
Accumulates value starting with the last element and applying [operation] from right to left\n * to each element with
its index in the original array and current accumulator value.\n * \n * Returns `null` if the array is empty.\n * \n
 * @param [operation] function that takes the index of an element, the element itself and current accumulator value,\n
 * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.reduceRightIndexedOrNull(operation: (index: Int, ULong, acc: ULong) -> ULong): ULong? {\n  var
index = lastIndex\n  if (index < 0) return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n
accumulator = operation(index, get(index), accumulator)\n    --index\n  }\n  return accumulator\n}\n\n/**\n *
Accumulates value starting with the last element and applying [operation] from right to left\n * to each element with
its index in the original array and current accumulator value.\n * \n * Returns `null` if the array is empty.\n * \n
 * @param [operation] function that takes the index of an element, the element itself and current accumulator value,\n
 * and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.reduceRightIndexedOrNull(operation: (index: Int, UByte, acc: UByte) -> UByte): UByte? {\n  var
index = lastIndex\n  if (index < 0) return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n
accumulator = operation(index, get(index), accumulator)\n    --index\n  }\n  return accumulator\n}\n\n/**\n *
Accumulates value starting with the last element and applying [operation] from right to left\n * to each element with
its index in the original array and current accumulator value.\n * \n * Returns `null` if the array is empty.\n * \n
 * @param [operation] function that takes the index of an element, the element itself and current accumulator value,\n
 * and calculates the next accumulator value.\n * \n * @sample

```

@param [operation] function that takes the index of an element, the element itself and current accumulator value, and calculates the next accumulator value.

`samples.collections.Collections.Aggregates.reduceRightOrNull`

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.reduceRightIndexedOrNull(operation: (index: Int, UShort, acc: UShort) -> UShort): UShort? {\n  var
index = lastIndex\n  if (index < 0) return null\n  var accumulator = get(index--)\n  while (index >= 0) {\n
accumulator = operation(index, get(index), accumulator)\n    --index\n  }\n  return accumulator\n}\n\n/**\n *
Accumulates value starting with the last element and applying [operation] from right to left\n * to each element and
current accumulator value.\n * Returns `null` if the array is empty.\n * @param [operation] function that
takes an element and current accumulator value,\n * and calculates the next accumulator value.\n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun UIntArray.reduceRightOrNull(operation: (UInt, acc: UInt) -> UInt):
UInt? {\n  var index = lastIndex\n  if (index < 0) return null\n  var accumulator = get(index--)\n  while (index
>= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n  return accumulator\n}\n\n/**\n *
Accumulates value starting with the last element and applying [operation] from right to left\n * to each element and
current accumulator value.\n * Returns `null` if the array is empty.\n * @param [operation] function that
takes an element and current accumulator value,\n * and calculates the next accumulator value.\n * @sample
samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun ULongArray.reduceRightOrNull(operation: (ULong, acc: ULong) ->
ULong): ULong? {\n  var index = lastIndex\n  if (index < 0) return null\n  var accumulator = get(index--)\n
while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n  return
accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to
left\n * to each element and current accumulator value.\n * Returns `null` if the array is empty.\n * @param
[operation] function that takes an element and current accumulator value,\n * and calculates the next accumulator
value.\n * @sample samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun UByteArray.reduceRightOrNull(operation: (UByte, acc: UByte) ->
UByte): UByte? {\n  var index = lastIndex\n  if (index < 0) return null\n  var accumulator = get(index--)\n
while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n  return
accumulator\n}\n\n/**\n * Accumulates value starting with the last element and applying [operation] from right to
left\n * to each element and current accumulator value.\n * Returns `null` if the array is empty.\n * @param
[operation] function that takes an element and current accumulator value,\n * and calculates the next accumulator
value.\n * @sample samples.collections.Collections.Aggregates.reduceRightOrNull\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun UShortArray.reduceRightOrNull(operation: (UShort, acc: UShort) ->
UShort): UShort? {\n  var index = lastIndex\n  if (index < 0) return null\n  var accumulator = get(index--)\n
while (index >= 0) {\n    accumulator = operation(get(index--), accumulator)\n  }\n  return
accumulator\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying
[operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n *
Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the
previous value in resulting list.\n * @param [operation] function that takes current accumulator value and an
element, and calculates the next accumulator value.\n * @sample
samples.collections.Collections.Aggregates.runningFold\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UIntArray.runningFold(initial: R, operation: (acc: R, UInt) -> R): List<R> {\n  if (isEmpty()) return
listOf(initial)\n  val result = ArrayList<R>(size + 1).apply { add(initial) }\n  var accumulator = initial\n  for
```

```

(element in this) {\n    accumulator = operation(accumulator, element)\n    result.add(accumulator)\n } \n
return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying
[operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n
* Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the
previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an
element, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningFold\n
*/\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
ULongArray.runningFold(initial: R, operation: (acc: R, ULong) -> R): List<R> {\n    if (isEmpty()) return
listOf(initial)\n    val result = ArrayList<R>(size + 1).apply { add(initial) }\n    var accumulator = initial\n    for
(element in this) {\n        accumulator = operation(accumulator, element)\n        result.add(accumulator)\n    }\n
return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying
[operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n
* Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the
previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an
element, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningFold\n
*/\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UByteArray.runningFold(initial: R, operation: (acc: R, UByte) -> R): List<R> {\n    if (isEmpty()) return
listOf(initial)\n    val result = ArrayList<R>(size + 1).apply { add(initial) }\n    var accumulator = initial\n    for
(element in this) {\n        accumulator = operation(accumulator, element)\n        result.add(accumulator)\n    }\n
return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying
[operation] from left to right\n * to each element and current accumulator value that starts with [initial] value.\n * \n
* Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the
previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value and an
element, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningFold\n
*/\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UShortArray.runningFold(initial: R, operation: (acc: R, UShort) -> R): List<R> {\n    if (isEmpty()) return
listOf(initial)\n    val result = ArrayList<R>(size + 1).apply { add(initial) }\n    var accumulator = initial\n    for
(element in this) {\n        accumulator = operation(accumulator, element)\n        result.add(accumulator)\n    }\n
return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying
[operation] from left to right\n * to each element, its index in the original array and current accumulator value that
starts with [initial] value.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n *
otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that takes the
index of an element, current accumulator value\n * and the element itself, and calculates the next accumulator
value.\n * \n * @sample samples.collections.Collections.Aggregates.runningFold\n
*/\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>
UIntArray.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, UInt) -> R): List<R> {\n    if (isEmpty())
return listOf(initial)\n    val result = ArrayList<R>(size + 1).apply { add(initial) }\n    var accumulator = initial\n
for (index in indices) {\n        accumulator = operation(index, accumulator, this[index])\n        result.add(accumulator)\n    }\n
return result\n}\n\n/**\n * Returns a list containing successive accumulation
values generated by applying [operation] from left to right\n * to each element, its index in the original array and
current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed to [operation]
function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation]
function that takes the index of an element, current accumulator value\n * and the element itself, and calculates the
next accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningFold\n
*/\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R>

```

```

ULongArray.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, ULong) -> R): List<R> {
    if (isEmpty()) return listOf(initial)
    val result = ArrayList<R>(size + 1).apply { add(initial) }
    var accumulator = initial
    for (index in indices) {
        accumulator = operation(index, accumulator, this[index])
        result.add(accumulator)
    }
    return result
}

/**
 * Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original array and current accumulator value that starts with [initial] value.
 * Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.
 * @param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.
 * @sample samples.collections.Collections.Aggregates.runningFold
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun <R>
UByteArray.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, UByte) -> R): List<R> {
    if (isEmpty()) return listOf(initial)
    val result = ArrayList<R>(size + 1).apply { add(initial) }
    var accumulator = initial
    for (index in indices) {
        accumulator = operation(index, accumulator, this[index])
        result.add(accumulator)
    }
    return result
}

/**
 * Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element, its index in the original array and current accumulator value that starts with [initial] value.
 * Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.
 * @param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value.
 * @sample samples.collections.Collections.Aggregates.runningFold
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun <R>
UShortArray.runningFoldIndexed(initial: R, operation: (index: Int, acc: R, UShort) -> R): List<R> {
    if (isEmpty()) return listOf(initial)
    val result = ArrayList<R>(size + 1).apply { add(initial) }
    var accumulator = initial
    for (index in indices) {
        accumulator = operation(index, accumulator, this[index])
        result.add(accumulator)
    }
    return result
}

/**
 * Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with the first element of this array.
 * Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.
 * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.
 * @sample samples.collections.Collections.Aggregates.runningReduce
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UIntArray.runningReduce(operation: (acc: UInt, UInt) -> UInt): List<UInt> {
    if (isEmpty()) return emptyList()
    var accumulator = this[0]
    val result = ArrayList<UInt>(size).apply { add(accumulator) }
    for (index in 1 until size) {
        accumulator = operation(accumulator, this[index])
        result.add(accumulator)
    }
    return result
}

/**
 * Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with the first element of this array.
 * Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.
 * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.
 * @sample samples.collections.Collections.Aggregates.runningReduce
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
ULongArray.runningReduce(operation: (acc: ULong, ULong) -> ULong): List<ULong> {
    if (isEmpty()) return emptyList()
    var accumulator = this[0]
    val result = ArrayList<ULong>(size).apply { add(accumulator) }
    for (index in 1 until size) {
        accumulator = operation(accumulator, this[index])
        result.add(accumulator)
    }
    return result
}

/**
 * Returns a list containing successive accumulation values generated by applying [operation] from left to right to each element and current accumulator value that starts with the first element of this array.
 * Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list.
 * @param [operation] function that takes current accumulator value and an element, and calculates the next accumulator value.
 * @sample
 */

```

```

samples.collections.Collections.Aggregates.runningReduce\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.runningReduce(operation: (acc: UByte, UByte) -> UByte): List<UByte> {\n  if (isEmpty()) return
emptyList()\n  var accumulator = this[0]\n  val result = ArrayList<UByte>(size).apply { add(accumulator) }\n
for (index in 1 until size) {\n    accumulator = operation(accumulator, this[index])\n    result.add(accumulator)\n
  }\n  return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying
[operation] from left to right\n * to each element and current accumulator value that starts with the first element of
this array.\n * \n * Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would
affect the previous value in resulting list.\n * \n * @param [operation] function that takes current accumulator value
and an element, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningReduce\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UShortArray.runningReduce(operation: (acc: UShort, UShort) -> UShort): List<UShort> {\n  if (isEmpty()) return
emptyList()\n  var accumulator = this[0]\n  val result = ArrayList<UShort>(size).apply { add(accumulator) }\n
for (index in 1 until size) {\n    accumulator = operation(accumulator, this[index])\n    result.add(accumulator)\n
  }\n  return result\n}\n\n/**\n * Returns a list containing successive accumulation values generated by applying
[operation] from left to right\n * to each element, its index in the original array and current accumulator value that
starts with the first element of this array.\n * \n * Note that `acc` value passed to [operation] function should not be
mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n * @param [operation] function that
takes the index of an element, current accumulator value\n * and the element itself, and calculates the next
accumulator value.\n * \n * @sample samples.collections.Collections.Aggregates.runningReduce\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.runningReduceIndexed(operation: (index: Int, acc: UInt, UInt) -> UInt): List<UInt> {\n  if (isEmpty())
return emptyList()\n  var accumulator = this[0]\n  val result = ArrayList<UInt>(size).apply { add(accumulator)
}\n  for (index in 1 until size) {\n    accumulator = operation(index, accumulator, this[index])\n
result.add(accumulator)\n  }\n  return result\n}\n\n/**\n * Returns a list containing successive accumulation
values generated by applying [operation] from left to right\n * to each element, its index in the original array and
current accumulator value that starts with the first element of this array.\n * \n * Note that `acc` value passed to
[operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n *
@param [operation] function that takes the index of an element, current accumulator value\n * and the element
itself, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningReduce\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.runningReduceIndexed(operation: (index: Int, acc: ULong, ULong) -> ULong): List<ULong> {\n  if
(isEmpty()) return emptyList()\n  var accumulator = this[0]\n  val result = ArrayList<ULong>(size).apply {
add(accumulator) }\n  for (index in 1 until size) {\n    accumulator = operation(index, accumulator, this[index])\n
result.add(accumulator)\n  }\n  return result\n}\n\n/**\n * Returns a list containing successive accumulation
values generated by applying [operation] from left to right\n * to each element, its index in the original array and
current accumulator value that starts with the first element of this array.\n * \n * Note that `acc` value passed to
[operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n * \n *
@param [operation] function that takes the index of an element, current accumulator value\n * and the element
itself, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.runningReduce\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.runningReduceIndexed(operation: (index: Int, acc: UByte, UByte) -> UByte): List<UByte> {\n  if
(isEmpty()) return emptyList()\n  var accumulator = this[0]\n  val result = ArrayList<UByte>(size).apply {
add(accumulator) }\n  for (index in 1 until size) {\n    accumulator = operation(index, accumulator, this[index])\n
result.add(accumulator)\n  }\n  return result\n}\n\n/**\n * Returns a list containing successive accumulation

```

values generated by applying [operation] from left to right to each element, its index in the original array and current accumulator value that starts with the first element of this array. Note that `acc` value passed to [operation] function should not be mutated; otherwise it would affect the previous value in resulting list. @param [operation] function that takes the index of an element, current accumulator value and the element itself, and calculates the next accumulator value. @sample

`samples.collections.Collections.Aggregates.runningReduce`

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun  
UShortArray.runningReduceIndexed(operation: (index: Int, acc: UShort, UShort) -> UShort): List<UShort> {\n if  
(isEmpty()) return emptyList()\n var accumulator = this[0]\n val result = ArrayList<UShort>(size).apply {\n add(accumulator) }\n for (index in 1 until size) {\n accumulator = operation(index, accumulator, this[index])\n result.add(accumulator)\n }\n return result\n}\n\n/**\n * Returns a list containing successive accumulation  
values generated by applying [operation] from left to right to each element and current accumulator value that  
starts with [initial] value. Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list. @param [operation] function that takes current  
accumulator value and an element, and calculates the next accumulator value. @sample
```

`samples.collections.Collections.Aggregates.scan`

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun <R> UIntArray.scan(initial: R, operation: (acc: R, UInt) -> R):  
List<R> {\n return runningFold(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation  
values generated by applying [operation] from left to right to each element and current accumulator value that  
starts with [initial] value. Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list. @param [operation] function that takes current  
accumulator value and an element, and calculates the next accumulator value. @sample
```

`samples.collections.Collections.Aggregates.scan`

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun <R> ULongArray.scan(initial: R, operation: (acc: R, ULong) -> R):  
List<R> {\n return runningFold(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation  
values generated by applying [operation] from left to right to each element and current accumulator value that  
starts with [initial] value. Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list. @param [operation] function that takes current  
accumulator value and an element, and calculates the next accumulator value. @sample
```

`samples.collections.Collections.Aggregates.scan`

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun <R> UByteArray.scan(initial: R, operation: (acc: R, UByte) -> R):  
List<R> {\n return runningFold(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation  
values generated by applying [operation] from left to right to each element and current accumulator value that  
starts with [initial] value. Note that `acc` value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list. @param [operation] function that takes current  
accumulator value and an element, and calculates the next accumulator value. @sample
```

`samples.collections.Collections.Aggregates.scan`

```
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun <R> UShortArray.scan(initial: R, operation: (acc: R, UShort) -> R):  
List<R> {\n return runningFold(initial, operation)\n}\n\n/**\n * Returns a list containing successive accumulation  
values generated by applying [operation] from left to right to each element, its index in the original array and  
current accumulator value that starts with [initial] value. Note that `acc` value passed to [operation] function  
should not be mutated; otherwise it would affect the previous value in resulting list. @param [operation]  
function that takes the index of an element, current accumulator value and the element itself, and calculates the  
next accumulator value. @sample
```

`samples.collections.Collections.Aggregates.scan`

```

*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun <R> UIntArray.scanIndexed(initial: R, operation: (index: Int, acc: R,
UInt) -> R): List<R> {\n    return runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns a list containing
successive accumulation values generated by applying [operation] from left to right\n * to each element, its index in
the original array and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed
to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n *
\n * @param [operation] function that takes the index of an element, current accumulator value\n * and the element
itself, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun <R> ULongArray.scanIndexed(initial: R, operation: (index: Int, acc:
R, ULong) -> R): List<R> {\n    return runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns a list
containing successive accumulation values generated by applying [operation] from left to right\n * to each element,
its index in the original array and current accumulator value that starts with [initial] value.\n * \n * Note that `acc`
value passed to [operation] function should not be mutated;\n * otherwise it would affect the previous value in
resulting list.\n * \n * @param [operation] function that takes the index of an element, current accumulator value\n *
and the element itself, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun <R> UByteArray.scanIndexed(initial: R, operation: (index: Int, acc:
R, UByte) -> R): List<R> {\n    return runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns a list containing
successive accumulation values generated by applying [operation] from left to right\n * to each element, its index in
the original array and current accumulator value that starts with [initial] value.\n * \n * Note that `acc` value passed
to [operation] function should not be mutated;\n * otherwise it would affect the previous value in resulting list.\n *
\n * @param [operation] function that takes the index of an element, current accumulator value\n * and the element
itself, and calculates the next accumulator value.\n * \n * @sample
samples.collections.Collections.Aggregates.scan\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalStdlibApi::class)\n
@kotlin.internal.InlineOnly\npublic inline fun <R> UShortArray.scanIndexed(initial: R, operation: (index: Int, acc:
R, UShort) -> R): List<R> {\n    return runningFoldIndexed(initial, operation)\n}\n\n/**\n * Returns the sum of all
values produced by [selector] function applied to each element in the array.\n * \n * @Deprecated("Use sumOf
instead.", ReplaceWith("this.sumOf(selector)"))\n * @DeprecatedSinceKotlin(warningSince =
"1.5")\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UIntArray.sumBy(selector: (UInt) -> UInt): UInt {\n    var sum: UInt = 0u\n    for (element in this) {\n        sum +=
selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n * \n * @Deprecated("Use sumOf instead.",
ReplaceWith("this.sumOf(selector)"))\n * @DeprecatedSinceKotlin(warningSince =
"1.5")\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
ULongArray.sumBy(selector: (ULong) -> UInt): UInt {\n    var sum: UInt = 0u\n    for (element in this) {\n        sum
+= selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the array.\n * \n * @Deprecated("Use sumOf instead.",
ReplaceWith("this.sumOf(selector)"))\n * @DeprecatedSinceKotlin(warningSince =
"1.5")\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun
UByteArray.sumBy(selector: (UByte) -> UInt): UInt {\n    var sum: UInt = 0u\n    for (element in this) {\n        sum
+= selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector]
function applied to each element in the array.\n * \n * @Deprecated("Use sumOf instead.",
ReplaceWith("this.sumOf(selector)"))\n * @DeprecatedSinceKotlin(warningSince =
"1.5")\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun

```

```

UShortArray.sumBy(selector: (UShort) -> UInt): UInt {
    var sum: UInt = 0
    for (element in this) {
        sum += selector(element)
    }
    return sum
}
// Returns the sum of all values produced by [selector]
// function applied to each element in the array.
@Deprecated("Use sumOf instead.",
ReplaceWith("this.sumOf(selector)"))
@DeprecatedSinceKotlin(warningSince =
"1.5")
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UIntArray.sumByDouble(selector: (UInt) -> Double): Double {
    var sum: Double = 0.0
    for (element in this) {
        sum += selector(element)
    }
    return sum
}
// Returns the sum of all values produced by
// [selector] function applied to each element in the array.
@Deprecated("Use sumOf instead.",
ReplaceWith("this.sumOf(selector)"))
@DeprecatedSinceKotlin(warningSince =
"1.5")
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
ULongArray.sumByDouble(selector: (ULong) -> Double): Double {
    var sum: Double = 0.0
    for (element in this) {
        sum += selector(element)
    }
    return sum
}
// Returns the sum of all values produced by
// [selector] function applied to each element in the array.
@Deprecated("Use sumOf instead.",
ReplaceWith("this.sumOf(selector)"))
@DeprecatedSinceKotlin(warningSince =
"1.5")
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UByteArray.sumByDouble(selector: (UByte) -> Double): Double {
    var sum: Double = 0.0
    for (element in this) {
        sum += selector(element)
    }
    return sum
}
// Returns the sum of all values produced by
// [selector] function applied to each element in the array.
@Deprecated("Use sumOf instead.",
ReplaceWith("this.sumOf(selector)"))
@DeprecatedSinceKotlin(warningSince =
"1.5")
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UShortArray.sumByDouble(selector: (UShort) -> Double): Double {
    var sum: Double = 0.0
    for (element in this) {
        sum += selector(element)
    }
    return sum
}
// Returns the sum of all values produced by
// [selector] function applied to each element in the array.
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@Suppress("INAPPLICABLE_JVM_NAME")
@kotlin.jvm.JvmName("sumOfDouble")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun UIntArray.sumOf(selector:
(UInt) -> Double): Double {
    var sum: Double = 0.toDouble()
    for (element in this) {
        sum +=
selector(element)
    }
    return sum
}
// Returns the sum of all values produced by [selector] function
// applied to each element in the array.
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@Suppress("INAPPLICABLE_JVM_NAME")
@kotlin.jvm.JvmName("sumOfDouble")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun ULongArray.sumOf(selector:
(ULong) -> Double): Double {
    var sum: Double = 0.toDouble()
    for (element in this) {
        sum +=
selector(element)
    }
    return sum
}
// Returns the sum of all values produced by [selector] function
// applied to each element in the array.
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@Suppress("INAPPLICABLE_JVM_NAME")
@kotlin.jvm.JvmName("sumOfDouble")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun UByteArray.sumOf(selector:
(UByte) -> Double): Double {
    var sum: Double = 0.toDouble()
    for (element in this) {
        sum +=
selector(element)
    }
    return sum
}
// Returns the sum of all values produced by [selector] function
// applied to each element in the array.
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution
ByLambdaReturnType
@Suppress("INAPPLICABLE_JVM_NAME")
@kotlin.jvm.JvmName("sumOfDouble")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun UShortArray.sumOf(selector:
(UShort) -> Double): Double {
    var sum: Double = 0.toDouble()
    for (element in this) {
        sum +=
selector(element)
    }
    return sum
}
// Returns the sum of all values produced by [selector] function
// applied to each element in the array.
@SinceKotlin("1.4")
@OptIn(kotlin.experimental.ExperimentalTypeInference::class)
@OverloadResolution

```

```

ByLambdaReturnType\n@Suppress(\"INAPPLICABLE_JVM_NAME\")\n@kotlin.jvm.JvmName(\"sumOfInt\")\n
@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UIntArray.sumOf(selector: (UInt) -
> Int): Int {\n    var sum: Int = 0.toInt()\n    for (element in this) {\n        sum += selector(element)\n    }\n    return
sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in the
array.\n
*\n@SinceKotlin(\"1.4\")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@Suppress(\"INAPPLICABLE_JVM_NAME\")\n@kotlin.jvm.JvmName(\"sumOfInt\")\n
@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun ULongArray.sumOf(selector:
(ULong) -> Int): Int {\n    var sum: Int = 0.toInt()\n    for (element in this) {\n        sum += selector(element)\n    }\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in
the array.\n
*\n@SinceKotlin(\"1.4\")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@Suppress(\"INAPPLICABLE_JVM_NAME\")\n@kotlin.jvm.JvmName(\"sumOfInt\")\n
@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UByteArray.sumOf(selector:
(UByte) -> Int): Int {\n    var sum: Int = 0.toInt()\n    for (element in this) {\n        sum += selector(element)\n    }\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in
the array.\n
*\n@SinceKotlin(\"1.4\")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@Suppress(\"INAPPLICABLE_JVM_NAME\")\n@kotlin.jvm.JvmName(\"sumOfInt\")\n
@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UShortArray.sumOf(selector:
(UShort) -> Int): Int {\n    var sum: Int = 0.toInt()\n    for (element in this) {\n        sum += selector(element)\n    }\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in
the array.\n
*\n@SinceKotlin(\"1.4\")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@Suppress(\"INAPPLICABLE_JVM_NAME\")\n@kotlin.jvm.JvmName(\"sumOfLong\")
\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UIntArray.sumOf(selector: (UInt)
-> Long): Long {\n    var sum: Long = 0.toLong()\n    for (element in this) {\n        sum += selector(element)\n    }\n
return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function applied to each element in
the array.\n
*\n@SinceKotlin(\"1.4\")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@Suppress(\"INAPPLICABLE_JVM_NAME\")\n@kotlin.jvm.JvmName(\"sumOfLong\")
\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun ULongArray.sumOf(selector:
(ULong) -> Long): Long {\n    var sum: Long = 0.toLong()\n    for (element in this) {\n        sum +=
selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n
*\n@SinceKotlin(\"1.4\")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@Suppress(\"INAPPLICABLE_JVM_NAME\")\n@kotlin.jvm.JvmName(\"sumOfLong\")
\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UByteArray.sumOf(selector:
(UByte) -> Long): Long {\n    var sum: Long = 0.toLong()\n    for (element in this) {\n        sum +=
selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n
*\n@SinceKotlin(\"1.4\")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution
ByLambdaReturnType\n@Suppress(\"INAPPLICABLE_JVM_NAME\")\n@kotlin.jvm.JvmName(\"sumOfLong\")
\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun UShortArray.sumOf(selector:
(UShort) -> Long): Long {\n    var sum: Long = 0.toLong()\n    for (element in this) {\n        sum +=
selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values produced by [selector] function
applied to each element in the array.\n
*\n@SinceKotlin(\"1.5\")\n@OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n@OverloadResolution

```

```

ByLambdaReturnType\n@Suppress(\"INAPPLICABLE_JVM_NAME\")\n@kotlin.jvm.JvmName(\"sumOfUInt\")\n\n@ExperimentalUnsignedTypes\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.Inline\nOnly\npublic inline fun UIntArray.sumOf(selector: (UInt) -> UInt): UInt {\n    var sum: UInt = 0.toUInt()\n    for\n    (element in this) {\n        sum += selector(element)\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all values\n    produced by [selector] function applied to each element in the array.\n\n    *\n    @SinceKotlin(\"1.5\")\n    @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n    @OverloadResolution\n    ByLambdaReturnType\n    @Suppress(\"INAPPLICABLE_JVM_NAME\")\n    @kotlin.jvm.JvmName(\"sumOfUInt\")\n    @ExperimentalUnsignedTypes\n    @WasExperimental(ExperimentalUnsignedTypes::class)\n    @kotlin.internal.Inline\n    Only\n    public inline fun ULongArray.sumOf(selector: (ULong) -> UInt): UInt {\n        var sum: UInt = 0.toUInt()\n        for\n        (element in this) {\n            sum += selector(element)\n        }\n        return sum\n    }\n\n    /**\n     * Returns the sum of all\n        values produced by [selector] function applied to each element in the array.\n\n        *\n        @SinceKotlin(\"1.5\")\n        @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n        @OverloadResolution\n        ByLambdaReturnType\n        @Suppress(\"INAPPLICABLE_JVM_NAME\")\n        @kotlin.jvm.JvmName(\"sumOfUInt\")\n        @ExperimentalUnsignedTypes\n        @WasExperimental(ExperimentalUnsignedTypes::class)\n        @kotlin.internal.Inline\n        Only\n        public inline fun UByteArray.sumOf(selector: (UByte) -> UInt): UInt {\n            var sum: UInt = 0.toUInt()\n            for\n            (element in this) {\n                sum += selector(element)\n            }\n            return sum\n        }\n\n        /**\n         * Returns the sum of all\n            values produced by [selector] function applied to each element in the array.\n\n            *\n            @SinceKotlin(\"1.5\")\n            @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n            @OverloadResolution\n            ByLambdaReturnType\n            @Suppress(\"INAPPLICABLE_JVM_NAME\")\n            @kotlin.jvm.JvmName(\"sumOfUInt\")\n            @ExperimentalUnsignedTypes\n            @WasExperimental(ExperimentalUnsignedTypes::class)\n            @kotlin.internal.Inline\n            Only\n            public inline fun UShortArray.sumOf(selector: (UShort) -> UInt): UInt {\n                var sum: UInt = 0.toUInt()\n                for\n                (element in this) {\n                    sum += selector(element)\n                }\n                return sum\n            }\n\n            /**\n             * Returns the sum of all\n                values produced by [selector] function applied to each element in the array.\n\n                *\n                @SinceKotlin(\"1.5\")\n                @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n                @OverloadResolution\n                ByLambdaReturnType\n                @Suppress(\"INAPPLICABLE_JVM_NAME\")\n                @kotlin.jvm.JvmName(\"sumOfULong\")\n                @ExperimentalUnsignedTypes\n                @WasExperimental(ExperimentalUnsignedTypes::class)\n                @kotlin.internal.Inline\n                Only\n                public inline fun UIntArray.sumOf(selector: (UInt) -> ULong): ULong {\n                    var sum: ULong =\n                    0.toULong()\n                    for\n                    (element in this) {\n                        sum += selector(element)\n                    }\n                    return sum\n                }\n\n                /**\n                 * Returns\n                    the sum of all values produced by [selector] function applied to each element in the array.\n\n                    *\n                    @SinceKotlin(\"1.5\")\n                    @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n                    @OverloadResolution\n                    ByLambdaReturnType\n                    @Suppress(\"INAPPLICABLE_JVM_NAME\")\n                    @kotlin.jvm.JvmName(\"sumOfULong\")\n                    @ExperimentalUnsignedTypes\n                    @WasExperimental(ExperimentalUnsignedTypes::class)\n                    @kotlin.internal.Inline\n                    Only\n                    public inline fun ULongArray.sumOf(selector: (ULong) -> ULong): ULong {\n                        var sum: ULong =\n                        0.toULong()\n                        for\n                        (element in this) {\n                            sum += selector(element)\n                        }\n                        return sum\n                    }\n\n                    /**\n                     * Returns\n                        the sum of all values produced by [selector] function applied to each element in the array.\n\n                        *\n                        @SinceKotlin(\"1.5\")\n                        @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n                        @OverloadResolution\n                        ByLambdaReturnType\n                        @Suppress(\"INAPPLICABLE_JVM_NAME\")\n                        @kotlin.jvm.JvmName(\"sumOfULong\")\n                        @ExperimentalUnsignedTypes\n                        @WasExperimental(ExperimentalUnsignedTypes::class)\n                        @kotlin.internal.Inline\n                        Only\n                        public inline fun UByteArray.sumOf(selector: (UByte) -> ULong): ULong {\n                            var sum: ULong =\n                            0.toULong()\n                            for\n                            (element in this) {\n                                sum += selector(element)\n                            }\n                            return sum\n                        }\n\n                        /**\n                         * Returns\n                            the sum of all values produced by [selector] function applied to each element in the array.\n\n                            *\n                            @SinceKotlin(\"1.5\")\n                            @OptIn(kotlin.experimental.ExperimentalTypeInference::class)\n                            @OverloadResolution\n                            ByLambdaReturnType\n                            @Suppress(\"INAPPLICABLE_JVM_NAME\")\n                            @kotlin.jvm.JvmName(\"sumOfULong\")\n                            @ExperimentalUnsignedTypes\n                            @WasExperimental(ExperimentalUnsignedTypes::class)\n                            @kotlin.internal.Inline\n                            Only\n                            public inline fun UShortArray.sumOf(selector: (UShort) -> ULong): ULong {\n                                var sum: ULong =\n                                0.toULong()\n                                for\n                                (element in this) {\n                                    sum += selector(element)\n                                }\n                                return sum\n                            }\n\n                            /**\n                             * Returns a\n                                list of pairs built from the elements of `this` array and the [other] array with the same index.\n                                *\n                                * The returned list has\n                                    length of the shortest collection.\n                                *\n                                * @sample samples.collections.Iterables.Operations.zipIterable\n
```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic infix fun <R> UIntArray.zip(other: Array<out R>): List<Pair<UInt, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` array and the [other] array with the same index.\n * The returned list has length of the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic infix fun <R> ULongArray.zip(other: Array<out R>): List<Pair<ULong, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` array and the [other] array with the same index.\n * The returned list has length of the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic infix fun <R> UByteArray.zip(other: Array<out R>): List<Pair<UByte, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` array and the [other] array with the same index.\n * The returned list has length of the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic infix fun <R> UShortArray.zip(other: Array<out R>): List<Pair<UShort, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned list has length of the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, V> UIntArray.zip(other: Array<out R>, transform: (a: UInt, b: R) -> V): List<V> {\n    val size = minOf(size, other.size)\n    val list = ArrayList<V>(size)\n    for (i in 0 until size) {\n        list.add(transform(this[i], other[i]))\n    }\n    return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned list has length of the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, V> ULongArray.zip(other: Array<out R>, transform: (a: ULong, b: R) -> V): List<V> {\n    val size = minOf(size, other.size)\n    val list = ArrayList<V>(size)\n    for (i in 0 until size) {\n        list.add(transform(this[i], other[i]))\n    }\n    return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned list has length of the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, V> UByteArray.zip(other: Array<out R>, transform: (a: UByte, b: R) -> V): List<V> {\n    val size = minOf(size, other.size)\n    val list = ArrayList<V>(size)\n    for (i in 0 until size) {\n        list.add(transform(this[i], other[i]))\n    }\n    return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned list has length of the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, V> UShortArray.zip(other: Array<out R>, transform: (a: UShort, b: R) -> V): List<V> {\n    val size = minOf(size, other.size)\n    val list = ArrayList<V>(size)\n    for (i in 0 until size) {\n        list.add(transform(this[i], other[i]))\n    }\n    return list\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` collection and [other] array with the same index.\n * The returned list has length of the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic infix fun <R> UIntArray.zip(other: Iterable<R>): List<Pair<UInt, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built from the elements of `this` collection and [other] array with the same index.\n * The returned list has length of the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n

```

```

*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic infix fun <R> ULongArray.zip(other:
Iterable<R>): List<Pair<ULong, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs
built from the elements of `this` collection and [other] array with the same index.\n * The returned list has length of
the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic infix fun <R> UByteArray.zip(other:
Iterable<R>): List<Pair<UByte, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs
built from the elements of `this` collection and [other] array with the same index.\n * The returned list has length of
the shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic infix fun <R> UShortArray.zip(other:
Iterable<R>): List<Pair<UShort, R>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of
values built from the elements of `this` array and the [other] collection with the same index\n * using the provided
[transform] function applied to each pair of elements.\n * The returned list has length of the shortest collection.\n *
\n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, V>
UIntArray.zip(other: Iterable<R>, transform: (a: UInt, b: R) -> V): List<V> {\n    val arraySize = size\n    val list =
ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))\n    var i = 0\n    for (element in other) {\n
if (i >= arraySize) break\n        list.add(transform(this[i++], element))\n    }\n    return list\n}\n\n/**\n * Returns a
list of values built from the elements of `this` array and the [other] collection with the same index\n * using the
provided [transform] function applied to each pair of elements.\n * The returned list has length of the shortest
collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, V>
ULongArray.zip(other: Iterable<R>, transform: (a: ULong, b: R) -> V): List<V> {\n    val arraySize = size\n    val list =
ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))\n    var i = 0\n    for (element in other) {\n
if (i >= arraySize) break\n        list.add(transform(this[i++], element))\n    }\n    return list\n}\n\n/**\n * Returns a
list of values built from the elements of `this` array and the [other] collection with the same index\n * using the
provided [transform] function applied to each pair of elements.\n * The returned list has length of the shortest
collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, V>
UByteArray.zip(other: Iterable<R>, transform: (a: UByte, b: R) -> V): List<V> {\n    val arraySize = size\n    val list =
ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))\n    var i = 0\n    for (element in other) {\n
if (i >= arraySize) break\n        list.add(transform(this[i++], element))\n    }\n    return list\n}\n\n/**\n * Returns a
list of values built from the elements of `this` array and the [other] collection with the same index\n * using the
provided [transform] function applied to each pair of elements.\n * The returned list has length of the shortest
collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <R, V>
UShortArray.zip(other: Iterable<R>, transform: (a: UShort, b: R) -> V): List<V> {\n    val arraySize = size\n    val list =
ArrayList<V>(minOf(other.collectionSizeOrDefault(10), arraySize))\n    var i = 0\n    for (element in other) {\n
if (i >= arraySize) break\n        list.add(transform(this[i++], element))\n    }\n    return list\n}\n\n/**\n * Returns a
list of pairs built from the elements of `this` array and the [other] array with the same index.\n * The returned list has length of the shortest collection.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterable\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic infix fun UIntArray.zip(other: UIntArray):
List<Pair<UInt, UInt>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built from
the elements of `this` array and the [other] array with the same index.\n * The returned list has length of the shortest
collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic infix fun ULongArray.zip(other: ULongArray):
List<Pair<ULong, ULong>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built
from the elements of `this` array and the [other] array with the same index.\n * The returned list has length of the

```

```

shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic infix fun UByteArray.zip(other: UByteArray):
List<Pair<UByte, UByte>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of pairs built
from the elements of `this` array and the [other] array with the same index.\n * The returned list has length of the
shortest collection.\n * \n * @sample samples.collections.Iterables.Operations.zipIterable\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic infix fun UShortArray.zip(other: UShortArray):
List<Pair<UShort, UShort>> {\n    return zip(other) { t1, t2 -> t1 to t2 }\n}\n\n/**\n * Returns a list of values built
from the elements of `this` array and the [other] array with the same index\n * using the provided [transform]
function applied to each pair of elements.\n * The returned list has length of the shortest array.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <V>
UIntArray.zip(other: UIntArray, transform: (a: UInt, b: UInt) -> V): List<V> {\n    val size = minOf(size,
other.size)\n    val list = ArrayList<V>(size)\n    for (i in 0 until size) {\n        list.add(transform(this[i], other[i]))\n
    }\n    return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array
with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned
list has length of the shortest array.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <V>
ULongArray.zip(other: ULongArray, transform: (a: ULong, b: ULong) -> V): List<V> {\n    val size = minOf(size,
other.size)\n    val list = ArrayList<V>(size)\n    for (i in 0 until size) {\n        list.add(transform(this[i], other[i]))\n
    }\n    return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array
with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned
list has length of the shortest array.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <V>
UByteArray.zip(other: UByteArray, transform: (a: UByte, b: UByte) -> V): List<V> {\n    val size = minOf(size,
other.size)\n    val list = ArrayList<V>(size)\n    for (i in 0 until size) {\n        list.add(transform(this[i], other[i]))\n
    }\n    return list\n}\n\n/**\n * Returns a list of values built from the elements of `this` array and the [other] array
with the same index\n * using the provided [transform] function applied to each pair of elements.\n * The returned
list has length of the shortest array.\n * \n * @sample
samples.collections.Iterables.Operations.zipIterableWithTransform\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun <V>
UShortArray.zip(other: UShortArray, transform: (a: UShort, b: UShort) -> V): List<V> {\n    val size = minOf(size,
other.size)\n    val list = ArrayList<V>(size)\n    for (i in 0 until size) {\n        list.add(transform(this[i], other[i]))\n
    }\n    return list\n}\n\n/**\n * Returns the sum of all elements in the array.\n
*\n@kotlin.jvm.JvmName("sumOfUInt")\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedT
ypes::class)\npublic fun Array<out UInt>.sum(): UInt {\n    var sum: UInt = 0u\n    for (element in this) {\n        sum
+= element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in the array.\n
*\n@kotlin.jvm.JvmName("sumOfULong")\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsigned
Types::class)\npublic fun Array<out ULong>.sum(): ULong {\n    var sum: ULong = 0uL\n    for (element in this)
{\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in the array.\n
*\n@kotlin.jvm.JvmName("sumOfUByte")\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsigned
Types::class)\npublic fun Array<out UByte>.sum(): UInt {\n    var sum: UInt = 0u\n    for (element in this) {\n
sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in the array.\n
*\n@kotlin.jvm.JvmName("sumOfUShort")\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsigned
Types::class)\npublic fun Array<out UShort>.sum(): UInt {\n    var sum: UInt = 0u\n    for (element in this) {\n
sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in the array.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\n@kotlin.internal.InlineOnly\npublic inline fun

```

```

UIntArray.sum(): UInt {
    return storage.sum().toUInt()
}

/**
 * Returns the sum of all elements in the
 * array.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
ULongArray.sum(): ULong {
    return storage.sum().toULong()
}

/**
 * Returns the sum of all elements in
 * the array.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UByteArray.sum(): UInt {
    return sumOf { it.toUInt() }
}

/**
 * Returns the sum of all elements in the
 * array.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
@kotlin.internal.InlineOnly
public inline fun
UShortArray.sum(): UInt {
    return sumOf { it.toUInt() }
}

/**
 * Copyright 2010-2021 JetBrains s.r.o. and
 * Kotlin Programming Language contributors.
 * Use of this source code is governed by the Apache 2.0 license that
 * can be found in the license/LICENSE.txt file.
 */
@file:kotlin.jvm.JvmMultifileClass
@file:kotlin.jvm.JvmName("UCollectionsKt")
package
kotlin.collections

// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt
// See:
// https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib
import kotlin.random.*
import
kotlin.ranges.contains
import kotlin.ranges.reversed

/**
 * Returns an array of UByte containing all of the
 * elements of this collection.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
public fun
Collection<UByte>.toUByteArray(): UByteArray {
    val result = UByteArray(size)
    var index = 0
    for (element in this)
        result[index++] = element
    return result
}

/**
 * Returns an array of UInt containing
 * all of the elements of this collection.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
public fun
Collection<UInt>.toUIntArray(): UIntArray {
    val result = UIntArray(size)
    var index = 0
    for (element in this)
        result[index++] = element
    return result
}

/**
 * Returns an array of ULong containing all of the
 * elements of this collection.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
public fun
Collection<ULong>.toULongArray(): ULongArray {
    val result = ULongArray(size)
    var index = 0
    for (element in this)
        result[index++] = element
    return result
}

/**
 * Returns an array of UShort
 * containing all of the elements of this collection.
 */
@SinceKotlin("1.3")
@ExperimentalUnsignedTypes
public fun Collection<UShort>.toUShortArray():
UShortArray {
    val result = UShortArray(size)
    var index = 0
    for (element in this)
        result[index++] =
element
    return result
}

/**
 * Returns the sum of all elements in the collection.
 */
@kotlin.jvm.JvmName("sumOfUInt")
@SinceKotlin("1.5")
@WasExperimental(ExperimentalUnsignedT
ypes::class)
public fun Iterable<UInt>.sum(): UInt {
    var sum: UInt = 0u
    for (element in this) {
        sum
+= element
    }
    return sum
}

/**
 * Returns the sum of all elements in the collection.
 */
@kotlin.jvm.JvmName("sumOfULong")
@SinceKotlin("1.5")
@WasExperimental(ExperimentalUnsigned
Types::class)
public fun Iterable<ULong>.sum(): ULong {
    var sum: ULong = 0uL
    for (element in this) {
        sum
+= element
    }
    return sum
}

/**
 * Returns the sum of all elements in the collection.
 */
@kotlin.jvm.JvmName("sumOfUByte")
@SinceKotlin("1.5")
@WasExperimental(ExperimentalUnsigned
Types::class)
public fun Iterable<UByte>.sum(): UInt {
    var sum: UInt = 0u
    for (element in this) {
        sum
+= element
    }
    return sum
}

/**
 * Returns the sum of all elements in the collection.
 */
@kotlin.jvm.JvmName("sumOfUShort")
@SinceKotlin("1.5")
@WasExperimental(ExperimentalUnsigned
Types::class)
public fun Iterable<UShort>.sum(): UInt {
    var sum: UInt = 0u
    for (element in this) {
        sum
+= element
    }
    return sum
}

/**
 * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming
 * Language contributors.
 * Use of this source code is governed by the Apache 2.0 license that can be found in the
 * license/LICENSE.txt file.
 */
@file:kotlin.jvm.JvmMultifileClass
@file:kotlin.jvm.JvmName("UComparisonsKt")
package
kotlin.comparisons

// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt
// See:
// https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib
import kotlin.random.*

/**
 * Returns the
 * greater of two values.
 */
@SinceKotlin("1.5")
@WasExperimental(ExperimentalUnsignedTypes::class)
public fun maxOf(a: UInt, b:
UInt): UInt {
    return if (a >= b) a else b
}

/**
 * Returns the greater of two values.
 */
@SinceKotlin("1.5")
@WasExperimental(ExperimentalUnsignedTypes::class)
public fun maxOf(a: ULong,
b: ULong): ULong {
    return if (a >= b) a else b
}

```

```

*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun maxOf(a: UByte,
b: UByte): UByte {\n    return if (a >= b) a else b\n}\n\n/**\n * Returns the greater of two values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun maxOf(a: UShort,
b: UShort): UShort {\n    return if (a >= b) a else b\n}\n\n/**\n * Returns the greater of three values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\
npublic inline fun maxOf(a: UInt, b: UInt, c: UInt): UInt {\n    return maxOf(a, maxOf(b, c))\n}\n\n/**\n * Returns
the greater of three values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\
npublic inline fun maxOf(a: ULong, b: ULong, c: ULong): ULong {\n    return maxOf(a, maxOf(b, c))\n}\n\n/**\n *
Returns the greater of three values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\
npublic inline fun maxOf(a: UByte, b: UByte, c: UByte): UByte {\n    return maxOf(a, maxOf(b, c))\n}\n\n/**\n *
Returns the greater of three values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\
npublic inline fun maxOf(a: UShort, b: UShort, c: UShort): UShort {\n    return maxOf(a, maxOf(b, c))\n}\n\n/**\n
* Returns the greater of the given values.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun
maxOf(a: UInt, vararg other: UInt): UInt {\n    var max = a\n    for (e in other) max = maxOf(max, e)\n    return
max\n}\n\n/**\n * Returns the greater of the given values.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun maxOf(a: ULong, vararg other: ULong):
ULong {\n    var max = a\n    for (e in other) max = maxOf(max, e)\n    return max\n}\n\n/**\n * Returns the greater
of the given values.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun maxOf(a: UByte,
vararg other: UByte): UByte {\n    var max = a\n    for (e in other) max = maxOf(max, e)\n    return max\n}\n\n/**\n
* Returns the greater of the given values.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun
maxOf(a: UShort, vararg other: UShort): UShort {\n    var max = a\n    for (e in other) max = maxOf(max, e)\n
return max\n}\n\n/**\n * Returns the smaller of two values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun minOf(a: UInt, b:
UInt): UInt {\n    return if (a <= b) a else b\n}\n\n/**\n * Returns the smaller of two values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun minOf(a: ULong,
b: ULong): ULong {\n    return if (a <= b) a else b\n}\n\n/**\n * Returns the smaller of two values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun minOf(a: UByte,
b: UByte): UByte {\n    return if (a <= b) a else b\n}\n\n/**\n * Returns the smaller of two values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun minOf(a: UShort,
b: UShort): UShort {\n    return if (a <= b) a else b\n}\n\n/**\n * Returns the smaller of three values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\
npublic inline fun minOf(a: UInt, b: UInt, c: UInt): UInt {\n    return minOf(a, minOf(b, c))\n}\n\n/**\n * Returns
the smaller of three values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\
npublic inline fun minOf(a: ULong, b: ULong, c: ULong): ULong {\n    return minOf(a, minOf(b, c))\n}\n\n/**\n *
Returns the smaller of three values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\
npublic inline fun minOf(a: UByte, b: UByte, c: UByte): UByte {\n    return minOf(a, minOf(b, c))\n}\n\n/**\n *
Returns the smaller of three values.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\
npublic inline fun minOf(a: UShort, b: UShort, c: UShort): UShort {\n    return minOf(a, minOf(b, c))\n}\n\n/**\n
* Returns the smaller of the given values.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun
minOf(a: UInt, vararg other: UInt): UInt {\n    var min = a\n    for (e in other) min = minOf(min, e)\n    return
min\n}\n\n/**\n * Returns the smaller of the given values.\n
*\n@SinceKotlin("1.4")\n@ExperimentalUnsignedTypes\npublic fun minOf(a: ULong, vararg other: ULong):

```

```

ULong {
    var min = a
    for (e in other) min = minOf(min, e)
    return min
}

/**
 * Returns the smaller
of the given values.
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
public fun minOf(a: UByte,
vararg other: UByte): UByte {
    var min = a
    for (e in other) min = minOf(min, e)
    return min
}

/**
 * Returns the smaller of the given values.
 */
@SinceKotlin("1.4")
@ExperimentalUnsignedTypes
public fun
minOf(a: UShort, vararg other: UShort): UShort {
    var min = a
    for (e in other) min = minOf(min, e)
    return min
}

/**
 * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.
 * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.
 */
@file:kotlin.jvm.JvmMultifileClass
@file:kotlin.jvm.JvmName("URangesKt")
package
kotlin.ranges

// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt
// See:
// https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib
nimport kotlin.random.*

/**
 * Returns a
random element from this range.
 */
@throws IllegalArgumentException if this range is empty.
 */
@SinceKotlin("1.5")
@WasExperimental(ExperimentalUnsignedTypes::class)
@kotlin.internal.InlineOnly
public inline fun UIntRange.random(): UInt {
    return random(Random)
}

/**
 * Returns a random element
from this range.
 */
@throws IllegalArgumentException if this range is empty.
 */
@SinceKotlin("1.5")
@WasExperimental(ExperimentalUnsignedTypes::class)
@kotlin.internal.InlineOnly
public inline fun ULongRange.random(): ULong {
    return random(Random)
}

/**
 * Returns a random
element from this range using the specified source of randomness.
 */
@throws IllegalArgumentException if
this range is empty.
 */
@SinceKotlin("1.5")
@WasExperimental(ExperimentalUnsignedTypes::class)
public fun
UIntRange.random(random: Random): UInt {
    try {
        return random.nextUInt(this)
    } catch (e:
IllegalArgumentException) {
        throw NoSuchElementException(e.message)
    }
}

/**
 * Returns a
random element from this range using the specified source of randomness.
 */
@throws
IllegalArgumentException if this range is empty.
 */
@SinceKotlin("1.5")
@WasExperimental(ExperimentalUnsignedTypes::class)
public fun
ULongRange.random(random: Random): ULong {
    try {
        return random.nextULong(this)
    } catch (e:
IllegalArgumentException) {
        throw NoSuchElementException(e.message)
    }
}

/**
 * Returns a
random element from this range, or `null` if this range is empty.
 */
@SinceKotlin("1.5")
@WasExperimental(ExperimentalStdlibApi::class,
ExperimentalUnsignedTypes::class)
@kotlin.internal.InlineOnly
public inline fun UIntRange.randomOrNull():
UInt? {
    return randomOrNull(Random)
}

/**
 * Returns a random element from this range, or `null` if this
range is empty.
 */
@SinceKotlin("1.5")
@WasExperimental(ExperimentalStdlibApi::class,
ExperimentalUnsignedTypes::class)
@kotlin.internal.InlineOnly
public inline fun ULongRange.randomOrNull():
ULong? {
    return randomOrNull(Random)
}

/**
 * Returns a random element from this range using the
specified source of randomness, or `null` if this range is empty.
 */
@SinceKotlin("1.5")
@WasExperimental(ExperimentalStdlibApi::class,
ExperimentalUnsignedTypes::class)
public fun UIntRange.randomOrNull(random: Random): UInt? {
    if
(isEmpty())
        return null
    return random.nextUInt(this)
}

/**
 * Returns a random element from this
range using the specified source of randomness, or `null` if this range is empty.
 */
@SinceKotlin("1.5")
@WasExperimental(ExperimentalStdlibApi::class,
ExperimentalUnsignedTypes::class)
public fun ULongRange.randomOrNull(random: Random): ULong? {
    if
(isEmpty())
        return null
    return random.nextULong(this)
}

/**
 * Returns `true` if this range contains
the specified [element].
 */
Always returns `false` if the [element] is `null`.
 */
@SinceKotlin("1.5")
@WasExperimental(ExperimentalUnsignedTypes::class)
@kotlin.internal.InlineOnly
public inline operator fun UIntRange.contains(element: UInt?): Boolean {
    return element != null &&
contains(element)
}

/**
 * Returns `true` if this range contains the specified [element].
 */
Always returns
`false` if the [element] is `null`.
 */
@SinceKotlin("1.5")
@WasExperimental(ExperimentalUnsignedTypes::class)
@kotlin.internal.InlineOnly
public inline operator fun ULongRange.contains(element: ULong?): Boolean {
    return element != null &&
contains(element)
}

/**
 * Checks if the specified [value] belongs to this range.
 */

```

```

*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic operator fun
UIntRange.contains(value: UByte): Boolean {\n    return contains(value.toInt())\n}\n\n/**\n * Checks if the
specified [value] belongs to this range.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic operator fun
ULongRange.contains(value: UByte): Boolean {\n    return contains(value.toULong())\n}\n\n/**\n * Checks if the
specified [value] belongs to this range.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic operator fun
ULongRange.contains(value: UInt): Boolean {\n    return contains(value.toULong())\n}\n\n/**\n * Checks if the
specified [value] belongs to this range.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic operator fun
UIntRange.contains(value: ULong): Boolean {\n    return (value shr UInt.SIZE_BITS) == 0uL &&
contains(value.toInt())\n}\n\n/**\n * Checks if the specified [value] belongs to this range.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic operator fun
UIntRange.contains(value: UShort): Boolean {\n    return contains(value.toInt())\n}\n\n/**\n * Checks if the
specified [value] belongs to this range.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic operator fun
ULongRange.contains(value: UShort): Boolean {\n    return contains(value.toULong())\n}\n\n/**\n * Returns a
progression from this value down to the specified [to] value with the step -1.\n * \n * The [to] value should be less
than or equal to `this` value.\n * If the [to] value is greater than `this` value the returned progression is empty.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic infix fun
UByte.downTo(to: UByte): UIntProgression {\n    return UIntProgression.fromClosedRange(this.toInt(),
to.toInt(), -1)\n}\n\n/**\n * Returns a progression from this value down to the specified [to] value with the step -
1.\n * \n * The [to] value should be less than or equal to `this` value.\n * If the [to] value is greater than `this` value
the returned progression is empty.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic infix fun
UInt.downTo(to: UInt): UIntProgression {\n    return UIntProgression.fromClosedRange(this, to, -1)\n}\n\n/**\n *
Returns a progression from this value down to the specified [to] value with the step -1.\n * \n * The [to] value should
be less than or equal to `this` value.\n * If the [to] value is greater than `this` value the returned progression is
empty.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic infix fun
ULong.downTo(to: ULong): ULongProgression {\n    return ULongProgression.fromClosedRange(this, to, -
1L)\n}\n\n/**\n * Returns a progression from this value down to the specified [to] value with the step -1.\n * \n *
The [to] value should be less than or equal to `this` value.\n * If the [to] value is greater than `this` value the
returned progression is empty.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic infix fun
UShort.downTo(to: UShort): UIntProgression {\n    return UIntProgression.fromClosedRange(this.toInt(),
to.toInt(), -1)\n}\n\n/**\n * Returns a progression that goes over the same range in the opposite direction with the
same step.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
UIntProgression.reversed(): UIntProgression {\n    return UIntProgression.fromClosedRange(last, first, -
step)\n}\n\n/**\n * Returns a progression that goes over the same range in the opposite direction with the same
step.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
ULongProgression.reversed(): ULongProgression {\n    return ULongProgression.fromClosedRange(last, first, -
step)\n}\n\n/**\n * Returns a progression that goes over the same range with the given step.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic infix fun
UIntProgression.step(step: Int): UIntProgression {\n    checkStepIsPositive(step > 0, step)\n    return
UIntProgression.fromClosedRange(first, last, if (this.step > 0) step else -step)\n}\n\n/**\n * Returns a progression
that goes over the same range with the given step.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic infix fun
ULongProgression.step(step: Long): ULongProgression {\n    checkStepIsPositive(step > 0, step)\n    return

```

ULongProgression.fromClosedRange(first, last, if (this.step > 0) step else -step)\n\n\n\*\*\n \* Returns a range from this value up to but excluding the specified [to] value.\n \* \n \* If the [to] value is less than or equal to `this` value, then the returned range is empty.\n

```

*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic infix fun
UByte.until(to: UByte): UIntRange {\n  if (to <= UByte.MIN_VALUE) return UIntRange.EMPTY\n  return
this.toUInt() .. (to - 1u).toUInt()\n}\n\n**\n * Returns a range from this value up to but excluding the specified [to]
value.\n * \n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic infix fun UInt.until(to:
UInt): UIntRange {\n  if (to <= UInt.MIN_VALUE) return UIntRange.EMPTY\n  return this .. (to -
1u).toUInt()\n}\n\n**\n * Returns a range from this value up to but excluding the specified [to] value.\n * \n * If the
[to] value is less than or equal to `this` value, then the returned range is empty.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic infix fun
ULong.until(to: ULong): ULongRange {\n  if (to <= ULong.MIN_VALUE) return ULongRange.EMPTY\n
return this .. (to - 1u).toULong()\n}\n\n**\n * Returns a range from this value up to but excluding the specified [to]
value.\n * \n * If the [to] value is less than or equal to `this` value, then the returned range is empty.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic infix fun
UShort.until(to: UShort): UIntRange {\n  if (to <= UShort.MIN_VALUE) return UIntRange.EMPTY\n  return
this.toUInt() .. (to - 1u).toUInt()\n}\n\n**\n * Ensures that this value is not less than the specified
[minimumValue].\n * \n * @return this value if it's greater than or equal to the [minimumValue] or the
[minimumValue] otherwise.\n * \n * @sample samples.comparisons.ComparableOps.coerceAtLeastUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
UInt.coerceAtLeast(minimumValue: UInt): UInt {\n  return if (this < minimumValue) minimumValue else
this\n}\n\n**\n * Ensures that this value is not less than the specified [minimumValue].\n * \n * @return this value
if it's greater than or equal to the [minimumValue] or the [minimumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtLeastUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
ULong.coerceAtLeast(minimumValue: ULong): ULong {\n  return if (this < minimumValue) minimumValue else
this\n}\n\n**\n * Ensures that this value is not less than the specified [minimumValue].\n * \n * @return this value
if it's greater than or equal to the [minimumValue] or the [minimumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtLeastUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
UByte.coerceAtLeast(minimumValue: UByte): UByte {\n  return if (this < minimumValue) minimumValue else
this\n}\n\n**\n * Ensures that this value is not less than the specified [minimumValue].\n * \n * @return this value
if it's greater than or equal to the [minimumValue] or the [minimumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtLeastUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
UShort.coerceAtLeast(minimumValue: UShort): UShort {\n  return if (this < minimumValue) minimumValue else
this\n}\n\n**\n * Ensures that this value is not greater than the specified [maximumValue].\n * \n * @return this
value if it's less than or equal to the [maximumValue] or the [maximumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtMostUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
UInt.coerceAtMost(maximumValue: UInt): UInt {\n  return if (this > maximumValue) maximumValue else
this\n}\n\n**\n * Ensures that this value is not greater than the specified [maximumValue].\n * \n * @return this
value if it's less than or equal to the [maximumValue] or the [maximumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtMostUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
ULong.coerceAtMost(maximumValue: ULong): ULong {\n  return if (this > maximumValue) maximumValue else
this\n}\n\n**\n * Ensures that this value is not greater than the specified [maximumValue].\n * \n * @return this
value if it's less than or equal to the [maximumValue] or the [maximumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtMostUnsigned\n

```

```

value if it's less than or equal to the [maximumValue] or the [maximumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtMostUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
UByte.coerceAtMost(maximumValue: UByte): UByte {\n    return if (this > maximumValue) maximumValue else
this\n}\n\n/**\n * Ensures that this value is not greater than the specified [maximumValue].\n * \n * @return this
value if it's less than or equal to the [maximumValue] or the [maximumValue] otherwise.\n * \n * @sample
samples.comparisons.ComparableOps.coerceAtMostUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
UShort.coerceAtMost(maximumValue: UShort): UShort {\n    return if (this > maximumValue) maximumValue
else this\n}\n\n/**\n * Ensures that this value lies in the specified range [minimumValue]..[maximumValue].\n * \n
* @return this value if it's in the range, or [minimumValue] if this value is less than [minimumValue], or
[maximumValue] if this value is greater than [maximumValue].\n * \n * @sample
samples.comparisons.ComparableOps.coerceInUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
UInt.coerceIn(minimumValue: UInt, maximumValue: UInt): UInt {\n    if (minimumValue > maximumValue)
throw IllegalArgumentException("Cannot coerce value to an empty range: maximum $maximumValue is less than
minimum $minimumValue.")\n    if (this < minimumValue) return minimumValue\n    if (this > maximumValue)
return maximumValue\n    return this\n}\n\n/**\n * Ensures that this value lies in the specified range
[minimumValue]..[maximumValue].\n * \n * @return this value if it's in the range, or [minimumValue] if this value
is less than [minimumValue], or [maximumValue] if this value is greater than [maximumValue].\n * \n * @sample
samples.comparisons.ComparableOps.coerceInUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
ULong.coerceIn(minimumValue: ULong, maximumValue: ULong): ULong {\n    if (minimumValue >
maximumValue) throw IllegalArgumentException("Cannot coerce value to an empty range: maximum
$maximumValue is less than minimum $minimumValue.")\n    if (this < minimumValue) return minimumValue\n
if (this > maximumValue) return maximumValue\n    return this\n}\n\n/**\n * Ensures that this value lies in the
specified range [minimumValue]..[maximumValue].\n * \n * @return this value if it's in the range, or
[minimumValue] if this value is less than [minimumValue], or [maximumValue] if this value is greater than
[maximumValue].\n * \n * @sample samples.comparisons.ComparableOps.coerceInUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
UByte.coerceIn(minimumValue: UByte, maximumValue: UByte): UByte {\n    if (minimumValue >
maximumValue) throw IllegalArgumentException("Cannot coerce value to an empty range: maximum
$maximumValue is less than minimum $minimumValue.")\n    if (this < minimumValue) return minimumValue\n
if (this > maximumValue) return maximumValue\n    return this\n}\n\n/**\n * Ensures that this value lies in the
specified range [minimumValue]..[maximumValue].\n * \n * @return this value if it's in the range, or
[minimumValue] if this value is less than [minimumValue], or [maximumValue] if this value is greater than
[maximumValue].\n * \n * @sample samples.comparisons.ComparableOps.coerceInUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
UShort.coerceIn(minimumValue: UShort, maximumValue: UShort): UShort {\n    if (minimumValue >
maximumValue) throw IllegalArgumentException("Cannot coerce value to an empty range: maximum
$maximumValue is less than minimum $minimumValue.")\n    if (this < minimumValue) return minimumValue\n
if (this > maximumValue) return maximumValue\n    return this\n}\n\n/**\n * Ensures that this value lies in the
specified [range].\n * \n * @return this value if it's in the [range], or `range.start` if this value is less than
`range.start`, or `range.endInclusive` if this value is greater than `range.endInclusive`.\n * \n * @sample
samples.comparisons.ComparableOps.coerceInUnsigned\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
UInt.coerceIn(range: ClosedRange<UInt>): UInt {\n    if (range is ClosedFloatingPointRange) {\n        return
this.coerceIn<UInt>(range)\n    }\n    if (range.isEmpty()) throw IllegalArgumentException("Cannot coerce value to

```

```

an empty range: $range.)\n    return when {\n        this < range.start -> range.start\n        this > range.endInclusive -> range.endInclusive\n        else -> this\n    }\n}\n\n/**\n * Ensures that this value lies in the specified [range].\n * @return this value if it's in the [range], or `range.start` if this value is less than `range.start`, or `range.endInclusive` if this value is greater than `range.endInclusive`.\n * \n * @sample\n samples.comparisons.ComparableOps.coerceInUnsigned\n\n*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun\n ULong.coerceIn(range: ClosedRange<ULong>): ULong {\n    if (range is ClosedFloatingPointRange) {\n        return this.coerceIn<ULong>(range)\n    }\n    if (range.isEmpty()) throw IllegalArgumentException("Cannot coerce value to an empty range: $range.")\n    return when {\n        this < range.start -> range.start\n        this > range.endInclusive -> range.endInclusive\n        else -> this\n    }\n}\n\n"/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n\n*\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("USequencesKt")\n\npackage\n kotlin.sequences\n\n/\n\n// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt\n// See: https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\nimport kotlin.random.*\n\n/**\n * Returns the sum of all elements in the sequence.\n * \n * The operation is _terminal_.\n\n*\n@kotlin.jvm.JvmName("sumOfUInt")\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun Sequence<UInt>.sum(): UInt {\n    var sum: UInt = 0u\n    for (element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in the sequence.\n * \n * The operation is _terminal_.\n\n*\n@kotlin.jvm.JvmName("sumOfULong")\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun Sequence<ULong>.sum(): ULong {\n    var sum: ULong = 0uL\n    for (element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in the sequence.\n * \n * The operation is _terminal_.\n\n*\n@kotlin.jvm.JvmName("sumOfUByte")\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun Sequence<UByte>.sum(): UInt {\n    var sum: UInt = 0u\n    for (element in this) {\n        sum += element\n    }\n    return sum\n}\n\n/**\n * Returns the sum of all elements in the sequence.\n * \n * The operation is _terminal_.\n\n*\n@kotlin.jvm.JvmName("sumOfUShort")\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun Sequence<UShort>.sum(): UInt {\n    var sum: UInt = 0u\n    for (element in this) {\n        sum += element\n    }\n    return sum\n}\n\n"/*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n\n*\n\npackage kotlin\n\npublic expect open class Error : Throwable {\n    constructor()\n    constructor(message: String?)\n    constructor(message: String?, cause: Throwable?)\n\n    constructor(cause: Throwable?)\n}\n\npublic expect open class Exception : Throwable {\n    constructor()\n    constructor(message: String?)\n    constructor(message: String?, cause: Throwable?)\n    constructor(cause: Throwable?)\n}\n\npublic expect open class RuntimeException : Exception {\n    constructor()\n    constructor(message: String?)\n    constructor(message: String?, cause: Throwable?)\n    constructor(cause: Throwable?)\n}\n\npublic expect open class IllegalArgumentException : RuntimeException {\n    constructor()\n    constructor(message: String?)\n    constructor(message: String?, cause: Throwable?)\n    constructor(cause: Throwable?)\n}\n\npublic expect open class IllegalStateException : RuntimeException {\n    constructor()\n    constructor(message: String?)\n    constructor(message: String?, cause: Throwable?)\n    constructor(cause: Throwable?)\n}\n\npublic expect open class IndexOutOfBoundsException : RuntimeException {\n    constructor()\n    constructor(message: String?)\n}\n\npublic expect open class ConcurrentModificationException : RuntimeException {\n    constructor()\n    constructor(message: String?)\n\n    @Deprecated("The constructor is not supported on all platforms and will be removed from kotlin-stdlib-common soon.", level =\n        DeprecationLevel.ERROR)\n    constructor(message: String?, cause: Throwable?)\n\n    @Deprecated("The constructor is not supported on all platforms and will be removed from kotlin-stdlib-common soon.", level =

```



```

Kotlin("1.4")\npublic annotation class ExperimentalJsExport\n\n/**\n * Exports top-level declaration on JS\n platform.\n *\n * Compiled module exposes declarations that are marked with this annotation without name\n mangling.\n *\n * This annotation can be applied to either files or top-level declarations.\n *\n * It is currently\n prohibited to export the following kinds of declarations:\n *\n * * `expect` declarations\n * * inline functions with\n reified type parameters\n * * suspend functions\n * * secondary constructors without `@JsName`\n * * extension properties\n * * enum classes\n * * annotation classes\n *\n * Signatures of exported declarations must\n only contain "exportable" types:\n *\n * * `dynamic`, `Any`, `String`, `Boolean`, `Byte`, `Short`, `Int`, `Float`,\n `Double`\n * * `BooleanArray`, `ByteArray`, `ShortArray`, `IntArray`, `FloatArray`, `DoubleArray`\n * *\n `Array<exportable-type>`\n * * Function types with exportable parameters and return types\n * * `external` or\n `@JsExport` classes and interfaces\n * * Nullable counterparts of types above\n * * Unit return type. Must not be\n nullable\n *\n * This annotation is experimental, meaning that restrictions mentioned above are subject to change.\n *\n */\n\n@ExperimentalJsExport\n@Retention(AnnotationRetention.BINARY)\n@Target(CLASS, PROPERTY,\n FUNCTION, FILE)\n@SinceKotlin("1.4")\n@OptionalExpectation\npublic expect annotation class\n JsExport(),"/**\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use\n of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n *\n */\n\npackage kotlin.io\n\n/** Prints the line separator to the standard output stream. */\n\npublic expect fun\n println()\n\n/** Prints the given [message] and the line separator to the standard output stream. */\n\npublic expect fun\n println(message: Any?)\n\n/** Prints the given [message] to the standard output stream. */\n\npublic expect fun\n print(message: Any?)\n\n/** Reads a line of input from the standard input stream and returns it,\n * or throws a\n [RuntimeException] if EOF has already been reached when [readLn] is called.\n *\n * LF or CRLF is treated as the\n line terminator. Line terminator is not included in the returned string.\n *\n * Currently this function is not supported\n in Kotlin/JS and throws [UnsupportedOperationException].\n *\n */\n\n@SinceKotlin("1.6")\n\npublic expect fun\n readLn(): String\n\n/** Reads a line of input from the standard input stream and returns it,\n * or return `null` if\n EOF has already been reached when [readLnOrNull] is called.\n *\n * LF or CRLF is treated as the line terminator.\n Line terminator is not included in the returned string.\n *\n * Currently this function is not supported in Kotlin/JS\n and throws [UnsupportedOperationException].\n *\n */\n\n@SinceKotlin("1.6")\n\npublic expect fun\n readLnOrNull():\n String?\n\ninternal class ReadAfterEOFException(message: String?) : RuntimeException(message)\n\n\ninternal\n expect interface Serializable\n","/**\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language\n contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the\n license/LICENSE.txt file.\n *\n */\n\npackage kotlin.collections\n\nimport kotlin.internal.PlatformDependent\n\n/**\n * Classes that inherit from this interface can be represented as a sequence of elements that can\n * be iterated over.\n *\n * @param T the type of element being iterated over. The iterator is covariant in its element type.\n *\n */\n\npublic interface\n Iterable<out T> {\n\n /**\n * Returns an iterator over the elements of this object.\n *\n */\n\n public operator fun\n iterator(): Iterator<T>\n}\n\n/**\n * Classes that inherit from this interface can be represented as a sequence of\n elements that can\n * be iterated over and that supports removing elements during iteration.\n *\n * @param T the type\n of element being iterated over. The mutable iterator is invariant in its element type.\n *\n */\n\npublic interface\n MutableIterable<out T> : Iterable<T> {\n\n /**\n * Returns an iterator over the elements of this sequence that\n supports removing elements during iteration.\n *\n */\n\n override fun\n iterator(): MutableIterator<T>\n}\n\n/**\n * A\n generic collection of elements. Methods in this interface support only read-only access to the collection;\n *\n * read/write access is supported through the [MutableCollection] interface.\n *\n * @param E the type of elements\n contained in the collection. The collection is covariant in its element type.\n *\n */\n\npublic interface\n Collection<out E> : Iterable<E> {\n\n // Query Operations\n\n /**\n * Returns the size of the collection.\n *\n */\n\n public val\n size:\n Int\n\n /**\n * Returns `true` if the collection is empty (contains no elements), `false` otherwise.\n *\n */\n\n public fun\n isEmpty(): Boolean\n\n /**\n * Checks if the specified element is contained in this collection.\n *\n */\n\n public operator fun\n contains(element: @UnsafeVariance E): Boolean\n\n override fun\n iterator():\n Iterator<E>\n\n // Bulk Operations\n\n /**\n * Checks if all elements in the specified collection are contained in\n this collection.\n *\n */\n\n public fun\n containsAll(elements: Collection<@UnsafeVariance E>): Boolean\n}\n\n/**\n * A\n generic collection of elements that supports adding and removing elements.\n *\n * @param E the type of

```

```

elements contained in the collection. The mutable collection is invariant in its element type.
public interface
MutableCollection<E> : Collection<E>, MutableIterable<E> {
    // Query Operations
    override fun iterator():
MutableIterable<E>
    // Modification Operations
    /**
     * Adds the specified element to the collection.
     *
     * @return `true` if the element has been added, `false` if the collection does not support duplicates
     * and the element is already contained in the collection.
     */
    public fun add(element: E): Boolean
    /**
     * Removes a single instance of the specified element from this
     * collection, if it is present.
     *
     * @return
     * `true` if the element has been successfully removed; `false` if it was not present in the collection.
     */
    public fun remove(element: E): Boolean
    // Bulk Modification Operations
    /**
     * Adds all of the elements of
     the specified collection to this collection.
     *
     * @return `true` if any of the specified elements was added to
     the collection, `false` if the collection was not modified.
     */
    public fun addAll(elements: Collection<E>):
Boolean
    /**
     * Removes all of this collection's elements that are also contained in the specified
     collection.
     *
     * @return `true` if any of the specified elements was removed from the collection, `false` if
     the collection was not modified.
     */
    public fun removeAll(elements: Collection<E>): Boolean
    /**
     * Retains only the elements in this collection that are contained in the specified collection.
     *
     * @return
     * `true` if any element was removed from the collection, `false` if the collection was not modified.
     */
    public fun retainAll(elements: Collection<E>): Boolean
    /**
     * Removes all elements from this collection.
     */
    public fun clear(): Unit
}
/**
 * A generic ordered collection of elements. Methods in this interface
 support only read-only access to the list;
 * read/write access is supported through the [MutableList] interface.
 *
 * @param E the type of elements contained in the list. The list is covariant in its element type.
 */
public interface
List<out E> : Collection<E> {
    // Query Operations
    override val size: Int
    override fun isEmpty():
Boolean
    override fun contains(element: @UnsafeVariance E): Boolean
    override fun iterator():
Iterator<E>
    // Bulk Operations
    override fun containsAll(elements: Collection<@UnsafeVariance E>):
Boolean
    // Positional Access Operations
    /**
     * Returns the element at the specified index in the list.
     */
    public operator fun get(index: Int): E
    // Search Operations
    /**
     * Returns the index of the first
     occurrence of the specified element in the list, or -1 if the specified
     * element is not contained in the list.
     */
    public fun indexOf(element: @UnsafeVariance E): Int
    /**
     * Returns the index of the last
     occurrence of the specified element in the list, or -1 if the specified
     * element is not contained in the list.
     */
    public fun lastIndexOf(element: @UnsafeVariance E): Int
    // List Iterators
    /**
     * Returns a list
     iterator over the elements in this list (in proper sequence).
     */
    public fun listIterator(): ListIterator<E>
    /**
     * Returns a list iterator over the elements in this list (in proper sequence), starting at the specified [index].
     */
    public fun listIterator(index: Int): ListIterator<E>
    // View
    /**
     * Returns a view of the portion
     of this list between the specified [fromIndex] (inclusive) and [toIndex] (exclusive).
     *
     * The returned list is backed
     by this list, so non-structural changes in the returned list are reflected in this list, and vice-versa.
     *
     * Structural changes in the base list make the behavior of the view undefined.
     */
    public fun
subList(fromIndex: Int, toIndex: Int): List<E>
}
/**
 * A generic ordered collection of elements that supports
 adding and removing elements.
 *
 * @param E the type of elements contained in the list. The mutable list is invariant
 in its element type.
 */
public interface MutableList<E> : List<E>, MutableCollection<E> {
    // Modification
Operations
    /**
     * Adds the specified element to the end of this list.
     *
     * @return `true` because the
     list is always modified as the result of this operation.
     */
    override fun add(element: E): Boolean
    override fun remove(element: E): Boolean
    // Bulk Modification Operations
    /**
     * Adds all of the
     elements of the specified collection to the end of this list.
     *
     * The elements are appended in the order they
     appear in the [elements] collection.
     *
     * @return `true` if the list was changed as the result of the
     operation.
     */
    override fun addAll(elements: Collection<E>): Boolean
    /**
     * Inserts all of the
     elements of the specified collection [elements] into this list at the specified [index].
     *
     * @return `true` if the
     list was changed as the result of the operation.
     */
    public fun addAll(index: Int, elements: Collection<E>):
Boolean
    override fun removeAll(elements: Collection<E>): Boolean
    override fun retainAll(elements:
Collection<E>): Boolean
    override fun clear(): Unit
    // Positional Access Operations
    /**
     * Replaces
     the element at the specified position in this list with the specified element.
     *
     * @return the element

```

```

previously at the specified position.\n
 *^ public operator fun set(index: Int, element: E): E\n
 /**\n
 * Inserts an element into the list at the specified [index].\n
 *^ public fun add(index: Int, element: E): Unit\n
 /**\n
 * Removes an element at the specified [index] from the list.\n
 *^ @return the element that has been removed.\n
 *^ public fun removeAt(index: Int): E\n
 // List Iterators\n
 override fun listIterator(): MutableListIterator<E>\n
 override fun listIterator(index: Int): MutableListIterator<E>\n
 // View\n
 override fun subList(fromIndex: Int, toIndex: Int): MutableList<E>\n
 } \n\n /**\n
 * A generic unordered collection of elements that does not support duplicate elements.\n
 * Methods in this interface support only read-only access to the set;\n
 * read/write access is supported through the [MutableSet] interface.\n
 * @param E the type of elements contained in the set. The set is covariant in its element type.\n
 *^ public interface Set<out E> : Collection<E> {\n
 // Query Operations\n
 override val size: Int\n
 override fun isEmpty(): Boolean\n
 override fun contains(element: @UnsafeVariance E): Boolean\n
 override fun iterator(): Iterator<E>\n
 // Bulk Operations\n
 override fun containsAll(elements: Collection<@UnsafeVariance E>): Boolean\n
 } \n\n /**\n
 * A generic unordered collection of elements that does not support duplicate elements, and supports\n
 * adding and removing elements.\n
 * @param E the type of elements contained in the set. The mutable set is invariant in its element type.\n
 *^ public interface MutableSet<E> : Set<E>, MutableCollection<E> {\n
 // Query Operations\n
 override fun iterator(): MutableIterator<E>\n
 // Modification Operations\n
 /**\n
 * Adds the specified element to the set.\n
 *^ @return `true` if the element has been added, `false` if the element is already contained in the set.\n
 *^ override fun add(element: E): Boolean\n
 // Bulk Modification Operations\n
 override fun addAll(elements: Collection<E>): Boolean\n
 override fun removeAll(elements: Collection<E>): Boolean\n
 override fun retainAll(elements: Collection<E>): Boolean\n
 override fun clear(): Unit\n
 } \n\n /**\n
 * A collection that holds pairs of objects (keys and values) and supports efficiently retrieving\n
 * the value corresponding to each key. Map keys are unique; the map holds only one value for each key.\n
 * Methods in this interface support only read-only access to the map; read-write access is supported through\n
 * the [MutableMap] interface.\n
 * @param K the type of map keys. The map is invariant in its key type, as it\n
 * can accept key as a parameter (of [containsKey] for example) and return it in [keys] set.\n
 * @param V the type of map values. The map is covariant in its value type.\n
 *^ public interface Map<K, out V> {\n
 // Query Operations\n
 /**\n
 * Returns the number of key/value pairs in the map.\n
 *^ public val size: Int\n
 /**\n
 * Returns `true` if the map is empty (contains no elements), `false` otherwise.\n
 *^ public fun isEmpty(): Boolean\n
 /**\n
 * Returns `true` if the map contains the specified [key].\n
 *^ public fun containsKey(key: K): Boolean\n
 /**\n
 * Returns `true` if the map maps one or more keys to the specified [value].\n
 *^ public fun containsValue(value: @UnsafeVariance V): Boolean\n
 /**\n
 * Returns the value corresponding to the given [key], or `null` if such a key is not present in the map.\n
 *^ public operator fun get(key: K): V?\n
 /**\n
 * Returns the value corresponding to the given [key], or [defaultValue] if such a key is not present in the map.\n
 *^ @since JDK 1.8\n
 *^ @SinceKotlin("1.1")\n
 *^ @PlatformDependent\n
 *^ public fun getOrDefault(key: K, defaultValue: @UnsafeVariance V): V {\n
 // See default implementation in JDK sources\n
 throw NotImplementedError()\n
 } \n\n // Views\n
 /**\n
 * Returns a read-only [Set] of all keys in this map.\n
 *^ public val keys: Set<K>\n
 /**\n
 * Returns a read-only [Collection] of all values in this map. Note that this collection may contain duplicate values.\n
 *^ public val values: Collection<V>\n
 /**\n
 * Returns a read-only [Set] of all key/value pairs in this map.\n
 *^ public val entries: Set<Map.Entry<K, V>>\n
 /**\n
 * Represents a key/value pair held by a [Map].\n
 *^ public interface Entry<out K, out V> {\n
 /**\n
 * Returns the key of this key/value pair.\n
 *^ public val key: K\n
 /**\n
 * Returns the value of this key/value pair.\n
 *^ public val value: V\n
 } \n\n /**\n
 * A modifiable collection that holds pairs of objects (keys and values) and supports efficiently retrieving\n
 * the value corresponding to each key. Map keys are unique; the map holds only one value for each key.\n
 * @param K the type of map keys. The map is invariant in its key type.\n
 * @param V the type of map values. The mutable map is invariant in its value type.\n
 *^ public interface MutableMap<K, V> : Map<K, V> {\n
 // Modification Operations\n
 /**\n
 * Associates the specified [value] with the specified [key] in the map.\n
 *^ @return the previous value associated with the key, or `null` if the key was not present in the map.\n
 *^ public fun

```

```

put(key: K, value: V): V? {
    /**
     * Removes the specified key and its corresponding value from this map.
     * @return the previous value associated with the key, or `null` if the key was not present in the map.
     */
    public fun remove(key: K): V? {
        /**
         * Removes the entry for the specified key only if it is mapped to the
         * specified value.
         * @return true if entry was removed
         */
        @SinceKotlin("1.1")
        @PlatformDependent
        public fun remove(key: K, value: V): Boolean {
            // See default implementation in
            // JDK sources
            return true
        }
        /**
         * Bulk Modification Operations
         */
        /**
         * Updates this map with
         * key/value pairs from the specified map [from].
         */
        public fun putAll(from: Map<out K, V>): Unit {
            /**
             * Removes all elements from this map.
             */
            public fun clear(): Unit {
                // Views
            }
            /**
             * Returns a
             * [MutableSet] of all keys in this map.
             */
            override val keys: MutableSet<K> {
                /**
                 * Returns a
                 * [MutableCollection] of all values in this map. Note that this collection may contain duplicate values.
                 */
                override val values: MutableCollection<V> {
                    /**
                     * Returns a [MutableSet] of all key/value pairs in this
                     * map.
                     */
                    override val entries: MutableSet<MutableMap.MutableEntry<K, V>> {
                        /**
                         * Represents a
                         * key/value pair held by a [MutableMap].
                         */
                        public interface MutableEntry<K, V> : Map.Entry<K, V> {
                            /**
                             * Changes the value associated with the key of this entry.
                             * @return the previous value
                             * corresponding to the key.
                             */
                            public fun setValue(newValue: V): V
                        }
                    }
                }
            }
        }
    }
}
}

/**
 * Copyright 2010-
 * 2021 JetBrains s.r.o. and Kotlin Programming Language contributors.
 * Use of this source code is governed by the
 * Apache 2.0 license that can be found in the license/LICENSE.txt file.
 */
// Auto-generated file. DO NOT
// EDIT!
package kotlin.collections {
    /**
     * An iterator over a sequence of values of type `Byte`.
     */
    public abstract class ByteIterator : Iterator<Byte> {
        override final fun next() = nextByte()
        /**
         * Returns the next value in
         * the sequence without boxing.
         */
        public abstract fun nextByte(): Byte
    }
    /**
     * An iterator over a sequence of
     * values of type `Char`.
     */
    public abstract class CharIterator : Iterator<Char> {
        override final fun next() =
            nextChar()
        /**
         * Returns the next value in the sequence without boxing.
         */
        public abstract fun nextChar():
            Char
    }
    /**
     * An iterator over a sequence of values of type `Short`.
     */
    public abstract class ShortIterator :
        Iterator<Short> {
        override final fun next() =
            nextShort()
        /**
         * Returns the next value in the sequence without
         * boxing.
         */
        public abstract fun nextShort():
            Short
    }
    /**
     * An iterator over a sequence of values of type `Int`.
     */
    public abstract class IntIterator :
        Iterator<Int> {
        override final fun next() =
            nextInt()
        /**
         * Returns the
         * next value in the sequence without boxing.
         */
        public abstract fun nextInt():
            Int
    }
    /**
     * An iterator over a
     * sequence of values of type `Long`.
     */
    public abstract class LongIterator :
        Iterator<Long> {
        override final fun
            next() =
            nextLong()
        /**
         * Returns the next value in the sequence without boxing.
         */
        public abstract fun
            nextLong():
            Long
    }
    /**
     * An iterator over a sequence of values of type `Float`.
     */
    public abstract class
        FloatIterator :
        Iterator<Float> {
        override final fun
            next() =
            nextFloat()
        /**
         * Returns the next value in the
         * sequence without boxing.
         */
        public abstract fun
            nextFloat():
            Float
    }
    /**
     * An iterator over a sequence of
     * values of type `Double`.
     */
    public abstract class
        DoubleIterator :
        Iterator<Double> {
        override final fun
            next() =
            nextDouble()
        /**
         * Returns the next value in the
         * sequence without boxing.
         */
        public abstract fun
            nextDouble():
            Double
    }
    /**
     * An iterator over a sequence of values of type `Boolean`.
     */
    public abstract class
        BooleanIterator :
        Iterator<Boolean> {
        override final fun
            next() =
            nextBoolean()
        /**
         * Returns the next value
         * in the sequence without boxing.
         */
        public abstract fun
            nextBoolean():
            Boolean
    }
    /**
     * Copyright 2010-
     * 2021 JetBrains s.r.o. and Kotlin Programming Language contributors.
     * Use of this source code is governed by the
     * Apache 2.0 license that can be found in the license/LICENSE.txt file.
     */
    // Auto-generated file. DO NOT
    // EDIT!
    package kotlin.ranges {
        /**
         * An iterator over a progression of values of type `Char`.
         * @property
         * step the number by which the value is incremented on each step.
         */
        internal class CharProgressionIterator(first:
            Char, last: Char, val step: Int) : CharIterator() {
            private val finalElement: Int = last.code
            private var hasNext:
                Boolean = if (step > 0)
                    first <= last
                else first >= last
            private var next: Int =
                if (hasNext) first.code
                else
                    finalElement
            override fun hasNext():
                Boolean = hasNext
            override fun nextChar():
                Char {
                val value
                    = next
                if (value ==
                    finalElement) {
                    if (!hasNext)
                        throw
                            kotlin.NoSuchElementException
                    hasNext =
                        false
                }
                else {
                    next +=
                        step
                }
                return value.toChar()
            }
        }
        /**
         * An iterator over a progression of values of type `Int`.
         * @property
         * step the number by which the value is
         * incremented on each step.
         */
        internal class IntProgressionIterator(first: Int, last: Int, val step: Int) : IntIterator()
    }
}

```

```

{\n private val finalElement: Int = last\n private var hasNext: Boolean = if (step > 0) first <= last else first >=
last\n private var next: Int = if (hasNext) first else finalElement\n\n override fun hasNext(): Boolean =
hasNext\n\n override fun nextInt(): Int {\n val value = next\n if (value == finalElement) {\n if
(!hasNext) throw kotlin.NoSuchElementException()\n hasNext = false\n }\n else {\n next +=
step\n }\n return value\n }\n}\n\n/**\n * An iterator over a progression of values of type `Long`.\n *
@property step the number by which the value is incremented on each step.\n */\ninternal class
LongProgressionIterator(first: Long, last: Long, val step: Long) : LongIterator() {\n private val finalElement: Long
= last\n private var hasNext: Boolean = if (step > 0) first <= last else first >= last\n private var next: Long = if
(hasNext) first else finalElement\n\n override fun hasNext(): Boolean = hasNext\n\n override fun nextLong():
Long {\n val value = next\n if (value == finalElement) {\n if (!hasNext) throw
kotlin.NoSuchElementException()\n hasNext = false\n }\n else {\n next += step\n }\n
return value\n }\n}\n\n"/**\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\n// Auto-generated file. DO NOT EDIT!\n\npackage kotlin.ranges\n\nimport
kotlin.internal.getProgressionLastElement\n\n/**\n * A progression of values of type `Char`.\n */\npublic open class
CharProgression\n internal constructor\n (\n start: Char,\n endInclusive: Char,\n step: Int\n
): Iterable<Char> {\n init {\n if (step == 0) throw kotlin.IllegalArgumentException("\Step must be non-
zero.")\n if (step == Int.MIN_VALUE) throw kotlin.IllegalArgumentException("\Step must be greater than
Int.MIN_VALUE to avoid overflow on negation.")\n }\n\n /**\n * The first element in the progression.\n
*\n * The last element in the progression.\n */\n public val first: Char = start\n\n /**\n * The last element in the progression.\n */\n public val last:
Char = getProgressionLastElement(start.code, endInclusive.code, step).toChar()\n\n /**\n * The step of the
progression.\n */\n public val step: Int = step\n\n override fun iterator(): CharIterator =
CharProgressionIterator(first, last, step)\n\n /**\n * Checks if the progression is empty.\n */\n * Progression
with a positive step is empty if its first element is greater than the last element.\n * Progression with a negative
step is empty if its first element is less than the last element.\n */\n public open fun isEmpty(): Boolean = if
(step > 0) first > last else first < last\n\n override fun equals(other: Any?): Boolean =\n other is
CharProgression && (isEmpty() && other.isEmpty()) ||\n first == other.first && last == other.last && step ==
other.step\n\n override fun hashCode(): Int =\n if (isEmpty()) -1 else (31 * (31 * first.code + last.code) +
step)\n\n override fun toString(): String = if (step > 0) "$first..$last step $step" else "$first downTo $last step ${-
step}"\n\n companion object {\n /**\n * Creates CharProgression within the specified bounds of a
closed range.\n */\n * The progression starts with the [rangeStart] value and goes toward the [rangeEnd]
value not excluding it, with the specified [step].\n * In order to go backwards the [step] must be negative.\n
*\n * [step] must be greater than `Int.MIN_VALUE` and not equal to zero.\n */\n public fun
fromClosedRange(rangeStart: Char, rangeEnd: Char, step: Int): CharProgression = CharProgression(rangeStart,
rangeEnd, step)\n }\n}\n\n/**\n * A progression of values of type `Int`.\n */\npublic open class IntProgression\n internal constructor\n (\n start: Int,\n endInclusive: Int,\n step: Int\n
): Iterable<Int> {\n init {\n if (step == 0) throw kotlin.IllegalArgumentException("\Step must be non-zero.")\n if (step ==
Int.MIN_VALUE) throw kotlin.IllegalArgumentException("\Step must be greater than Int.MIN_VALUE to avoid
overflow on negation.")\n }\n\n /**\n * The first element in the progression.\n */\n public val first: Int =
start\n\n /**\n * The last element in the progression.\n */\n public val last: Int =
getProgressionLastElement(start, endInclusive, step)\n\n /**\n * The step of the progression.\n */\n public
val step: Int = step\n\n override fun iterator(): IntIterator = IntProgressionIterator(first, last, step)\n\n /**\n *
Checks if the progression is empty.\n */\n * Progression with a positive step is empty if its first element is
greater than the last element.\n * Progression with a negative step is empty if its first element is less than the last
element.\n */\n public open fun isEmpty(): Boolean = if (step > 0) first > last else first < last\n\n override fun
equals(other: Any?): Boolean =\n other is IntProgression && (isEmpty() && other.isEmpty()) ||\n first ==
other.first && last == other.last && step == other.step\n\n override fun hashCode(): Int =\n if (isEmpty()) -1
else (31 * (31 * first + last) + step)\n\n override fun toString(): String = if (step > 0) "$first..$last step $step" else

```

```

    companion object {
        /**
         * Creates IntProgression within the
         * specified bounds of a closed range.
         * The progression starts with the [rangeStart] value and goes
         * toward the [rangeEnd] value not excluding it, with the specified [step].
         * In order to go backwards the [step]
         * must be negative.
         * [step] must be greater than `Int.MIN_VALUE` and not equal to zero.
         */
        public fun fromClosedRange(rangeStart: Int, rangeEnd: Int, step: Int): IntProgression =
            IntProgression(rangeStart, rangeEnd, step)
    }

    /**
     * A progression of values of type `Long`.
     */
    public open class LongProgression(
        internal constructor(
            start: Long,
            endInclusive: Long,
            step: Long
        ): Iterable<Long> {
        init {
            if (step == 0L) throw kotlin.IllegalArgumentException("Step
            must be non-zero.")
            if (step == Long.MIN_VALUE) throw kotlin.IllegalArgumentException("Step must be
            greater than Long.MIN_VALUE to avoid overflow on negation.")
        }

        /**
         * The first element in the
         * progression.
         */
        public val first: Long = start

        /**
         * The last element in the progression.
         */
        public val last: Long = getProgressionLastElement(start, endInclusive, step)

        /**
         * The step of the
         * progression.
         */
        public val step: Long = step

        override fun iterator(): LongIterator =
            LongProgressionIterator(first, last, step)

        /**
         * Checks if the progression is empty.
         * Progression with a positive step is empty if its first element is greater than the last element.
         * Progression with a
         * negative step is empty if its first element is less than the last element.
         */
        public open fun isEmpty(): Boolean
            = if (step > 0) first > last else first < last

        override fun equals(other: Any?): Boolean =
            other is
            LongProgression && (isEmpty() && other.isEmpty() ||
            first == other.first && last == other.last && step ==
            other.step)

        override fun hashCode(): Int =
            if (isEmpty()) -1 else (31 * (31 * (first xor (first ushr 32)) +
            (last xor (last ushr 32))) + (step xor (step ushr 32))).toInt()

        override fun toString(): String = if (step > 0)
            "$first..$last step $step" else "$first downTo $last step $(-step)"
    }

    companion object {
        /**
         * Creates LongProgression within the specified bounds of a closed range.
         * The progression starts
         * with the [rangeStart] value and goes toward the [rangeEnd] value not excluding it, with the specified [step].
         * In order to go backwards the [step] must be negative.
         * [step] must be greater than
         * `Long.MIN_VALUE` and not equal to zero.
         */
        public fun fromClosedRange(rangeStart: Long,
            rangeEnd: Long, step: Long): LongProgression = LongProgression(rangeStart, rangeEnd, step)
    }

    /**
     * Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming Language contributors.
     * Use of this source code is
     * governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.
     */
    package
    kotlin.ranges

    /**
     * Represents a range of values (for example, numbers or characters).
     * See the [Kotlin
     * language documentation](https://kotlinlang.org/docs/reference/ranges.html) for more information.
     */
    public interface ClosedRange<T>: Comparable<T> {
        /**
         * The minimum value in the range.
         */
        public val start: T

        /**
         * The maximum value in the range (inclusive).
         */
        public val endInclusive: T

        /**
         * Checks whether the specified [value] belongs to the range.
         */
        public operator fun contains(value:
            T): Boolean = value >= start && value <= endInclusive

        /**
         * Checks whether the range is empty.
         * The range is empty if its start value is greater than the end value.
         */
        public fun isEmpty(): Boolean = start
            > endInclusive
    }

    /**
     * Copyright 2010-2015 JetBrains s.r.o.
     * Licensed under the Apache License,
     * Version 2.0 (the "License");
     * you may not use this file except in compliance with the License.
     * You may
     * obtain a copy of the License at
     * http://www.apache.org/licenses/LICENSE-2.0
     * Unless required by
     * applicable law or agreed to in writing, software
     * distributed under the License is distributed on an "AS IS"
     * BASIS,
     * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
     * See the
     * License for the specific language governing permissions and
     * limitations under the License.
     */
    package
    kotlin

    /**
     * The type with only one value: the `Unit` object. This type corresponds to the `void` type in Java.
     */
    public object Unit {
        override fun toString() = "kotlin.Unit"
    }

    /**
     * Copyright 2010-2015 JetBrains
     * s.r.o.
     * Licensed under the Apache License, Version 2.0 (the "License");
     * you may not use this file except
     * in compliance with the License.
     * You may obtain a copy of the License at
     * http://www.apache.org/licenses/LICENSE-2.0
     * Unless required by applicable law or agreed to in writing,
     * software
     * distributed under the License is distributed on an "AS IS"
     * BASIS,
     * WITHOUT WARRANTIES
     * OR CONDITIONS OF ANY KIND, either express or implied.
     * See the License for the specific language
    
```

governing permissions and limitations under the License.

```

package kotlin.annotation
import kotlin.annotation.AnnotationTarget

/** Contains the list of code elements which are the possible annotation targets
public enum class AnnotationTarget {
    /** Class, interface or object, annotation class is also included
    CLASS,
    /** Annotation class only
    ANNOTATION_CLASS,
    /** Generic type parameter
    TYPE_PARAMETER,
    /** Property
    PROPERTY,
    /** Field, including property's backing field
    FIELD,
    /** Local variable
    LOCAL_VARIABLE,
    /** Value parameter of a function or a constructor
    VALUE_PARAMETER,
    /** Constructor only (primary or secondary)
    CONSTRUCTOR,
    /** Function (constructors are not included)
    FUNCTION,
    /** Property getter only
    PROPERTY_GETTER,
    /** Property setter only
    PROPERTY_SETTER,
    /** Type usage
    TYPE,
    /** Any expression
    EXPRESSION,
    /** File
    FILE,
    /** Type alias
    TYPEALIAS
}

```

`@SinceKotlin("1.1")` Contains the list of possible annotation's retentions. Determines how an annotation is stored in binary output.

```

public enum class AnnotationRetention {
    /** Annotation isn't stored in binary output
    SOURCE,
    /** Annotation is stored in binary output, but invisible for reflection
    BINARY,
    /** Annotation is stored in binary output and visible for reflection (default retention)
    RUNTIME
}

```

This meta-annotation indicates the kinds of code elements which are possible targets of an annotation. If the target meta-annotation is not present on an annotation declaration, the annotation is applicable to the following elements: [CLASS], [PROPERTY], [FIELD], [LOCAL\_VARIABLE], [VALUE\_PARAMETER], [CONSTRUCTOR], [FUNCTION], [PROPERTY\_GETTER], [PROPERTY\_SETTER].

`@property allowedTargets` list of allowed annotation targets

```

@Target(AnnotationTarget.ANNOTATION_CLASS)
@MustBeDocumented
public annotation class Target(
    vararg val allowedTargets: AnnotationTarget
)

```

This meta-annotation determines whether an annotation is stored in binary output and visible for reflection. By default, both are true.

`@property value necessary annotation retention (RUNTIME, BINARY or SOURCE)`

```

@Target(AnnotationTarget.ANNOTATION_CLASS)
public annotation class Retention(
    val value: AnnotationRetention = AnnotationRetention.RUNTIME
)

```

This meta-annotation determines that an annotation is applicable twice or more on a single code element

```

@Target(AnnotationTarget.ANNOTATION_CLASS)
public annotation class Repeatable

```

This meta-annotation determines that an annotation is a part of public API and therefore should be included in the generated documentation for the element to which the annotation is applied.

```

@Target(AnnotationTarget.ANNOTATION_CLASS)
public annotation class MustBeDocumented

```

Copyright 2010-2016 JetBrains s.r.o. Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0> Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

```

package kotlin.internal

```

Specifies that the corresponding type parameter is not used for unsafe operations such as casts or 'is' checks. That means it's completely safe to use generic types as argument for such parameter.

```

@Target(AnnotationTarget.TYPE_PARAMETER)
@Retention(AnnotationRetention.BINARY)
internal annotation class PureReifiable

```

Specifies that the corresponding built-in method exists depending on platform. Current implementation for JVM looks whether method with same JVM descriptor exists in the module JDK. For example `MutableMap.remove(K, V)` available only if corresponding method `'java.util.Map.remove(Ljava/lang/Object;Ljava/lang/Object;)Z'` is defined in JDK (i.e. for major versions  $\geq 8$ )

```

@Target(AnnotationTarget.FUNCTION)
@Retention(AnnotationRetention.BINARY)
internal annotation class PlatformDependent

```

Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors. Use of this source code is governed by the Apache 2.0 license that can be found in the `license/LICENSE.txt` file.

```

package kotlin.internal
// a mod b (in arithmetical sense)
private fun mod(a: Int, b: Int): Int {
    val mod = a % b
    return if (mod >= 0) mod else mod + b
}
private fun mod(a: Long, b:

```

```

Long): Long {
    val mod = a % b
    return if (mod >= 0) mod else mod + b
}

private fun
differenceModulo(a: Int, b: Int, c: Int): Int {
    return mod(mod(a, c) - mod(b, c), c)
}

private fun
differenceModulo(a: Long, b: Long, c: Long): Long {
    return mod(mod(a, c) - mod(b, c), c)
}

/**
 * Calculates the final element of a bounded arithmetic progression, i.e. the last element of the progression which is in
 * the range from [start] to [end] in case of a positive [step], or from [end] to [start] in case of a negative
 * [step].
 * No validation on passed parameters is performed. The given parameters should satisfy the
 * condition:
 * - either `step > 0` and `start <= end`,
 * - or `step < 0` and `start >= end`.
 * @param start first element of the progression
 * @param end ending bound for the progression
 * @param step increment, or
 * difference of successive elements in the progression
 * @return the final element of the progression
 */
@Suppress("ApiInternal")
internal fun getProgressionLastElement(start: Int, end: Int, step: Int): Int = when {
    step > 0 -> if (start >= end) end else end - differenceModulo(end, start, step)
    step < 0 -> if (start <= end)
        end else end + differenceModulo(start, end, -step)
    else -> throw kotlin.IllegalArgumentException("Step is
        zero.")
}

/**
 * Calculates the final element of a bounded arithmetic progression, i.e. the last element of the
 * progression which is in the range from [start] to [end] in case of a positive [step], or from [end] to [start] in case
 * of a negative [step].
 * No validation on passed parameters is performed. The given parameters should
 * satisfy the condition:
 * - either `step > 0` and `start <= end`,
 * - or `step < 0` and `start >= end`.
 * @param start first element of the progression
 * @param end ending bound for the progression
 * @param step
 * increment, or difference of successive elements in the progression
 * @return the final element of the progression
 */
@Suppress("ApiInternal")
internal fun getProgressionLastElement(start: Long, end: Long, step: Long):
    Long = when {
    step > 0 -> if (start >= end) end else end - differenceModulo(end, start, step)
    step < 0 -> if
        (start <= end) end else end + differenceModulo(start, end, -step)
    else -> throw
        kotlin.IllegalArgumentException("Step is zero.")
}

/* Copyright 2010-2018 JetBrains s.r.o. and Kotlin
 * Programming Language contributors.
 * Use of this source code is governed by the Apache 2.0 license that can be
 * found in the license/LICENSE.txt file.
 */
@JsName("arrayIterator")
internal fun arrayIterator(array:
    dynamic, type: String?) = when (type) {
    null -> {
        val arr: Array<dynamic> = array
        object :
            Iterator<dynamic> {
            var index = 0
            override fun hasNext() = index < arr.size
            override fun
                next() = if (index < arr.size) arr[index++] else throw NoSuchElementException("$index")
        }
    }
    "BooleanArray" -> booleanArrayIterator(array)
    "ByteArray" -> byteArrayIterator(array)
    "ShortArray" -
        > shortArrayIterator(array)
    "CharArray" -> charArrayIterator(array)
    "IntArray" ->
        intArrayIterator(array)
    "LongArray" -> longArrayIterator(array)
    "FloatArray" ->
        floatArrayIterator(array)
    "DoubleArray" -> doubleArrayIterator(array)
    else -> throw
        IllegalStateException("Unsupported type argument for arrayIterator:
            $type")
}

@JsName("booleanArrayIterator")
internal fun booleanArrayIterator(array: BooleanArray) =
    object : BooleanIterator() {
    var index = 0
    override fun hasNext() = index < array.size
    override fun
        nextBoolean() = if (index < array.size) array[index++] else throw
            NoSuchElementException("$index")
}

@JsName("byteArrayIterator")
internal fun byteArrayIterator(array:
    ByteArray) = object : ByteIterator() {
    var index = 0
    override fun hasNext() = index < array.size
    override fun
        nextByte() = if (index < array.size) array[index++] else throw
            NoSuchElementException("$index")
}

@JsName("shortArrayIterator")
internal fun
    shortArrayIterator(array: ShortArray) = object : ShortIterator() {
    var index = 0
    override fun hasNext() =
        index < array.size
    override fun nextShort() = if (index < array.size) array[index++] else throw
        NoSuchElementException("$index")
}

@JsName("charArrayIterator")
internal fun charArrayIterator(array:
    CharArray) = object : CharIterator() {
    var index = 0
    override fun hasNext() = index < array.size
    override fun
        nextChar() = if (index < array.size) array[index++] else throw
            NoSuchElementException("$index")
}

@JsName("intArrayIterator")
internal fun intArrayIterator(array:
    IntArray) = object : IntIterator() {
    var index = 0
    override fun hasNext() = index < array.size
    override fun
        nextInt() = if (index < array.size) array[index++] else throw
            NoSuchElementException("$index")
}

@JsName("floatArrayIterator")
internal fun

```

```

floatArrayIterator(array: FloatArray) = object : FloatIterator() {\n  var index = 0\n  override fun hasNext() = index
< array.size\n  override fun nextFloat() = if (index < array.size) array[index++] else throw
NoSuchElementException("\$index")\n}\n\n@JsName("doubleArrayIterator")\ninternal fun
doubleArrayIterator(array: DoubleArray) = object : DoubleIterator() {\n  var index = 0\n  override fun hasNext()
= index < array.size\n  override fun nextDouble() = if (index < array.size) array[index++] else throw
NoSuchElementException("\$index")\n}\n\n@JsName("longArrayIterator")\ninternal fun longArrayIterator(array:
LongArray) = object : LongIterator() {\n  var index = 0\n  override fun hasNext() = index < array.size\n
override fun nextLong() = if (index < array.size) array[index++] else throw
NoSuchElementException("\$index")\n}\n\n@JsName("PropertyMetadata")\ninternal class
PropertyMetadata(@JsName("callableName") val name:
String)\n\n@JsName("noWhenBranchMatched")\ninternal fun noWhenBranchMatched(): Nothing = throw
NoWhenBranchMatchedException()\n\n@JsName("subSequence")\ninternal fun subSequence(c: CharSequence,
startIndex: Int, endIndex: Int): CharSequence {\n  if (c is String) {\n    return c.substring(startIndex, endIndex)\n
} else {\n    return c.asDynamic().`subSequence_vux9f0$`(startIndex, endIndex)\n
}\n}\n\n@JsName("captureStack")\ninternal fun captureStack(@Suppress("UNUSED_PARAMETER")
baseClass: JsClass<in Throwable>, instance: Throwable) {\n  if (js("Error").captureStackTrace) {\n    // Using
uncropped stack traces due to KT-37563.\n    // Precise stack traces are implemented in JS IR compiler and
stdlib\n    js("Error").captureStackTrace(instance);\n  } else {\n    instance.asDynamic().stack = js("new
Error()").stack;\n  }\n}\n\n@JsName("newThrowable")\ninternal fun newThrowable(message: String?, cause:
Throwable?): Throwable {\n  val throwable = js("new Error()")\n  throwable.message = if (jsTypeOf(message)
== "undefined") {\n    if (cause != null) cause.toString() else null\n  } else {\n    message\n  }\n
throwable.cause = cause\n  throwable.name = "Throwable"\n  return
throwable\n}\n\n@JsName("BoxedChar")\ninternal class BoxedChar(val c: Int) : Comparable<Int> {\n  override
fun equals(other: Any?): Boolean {\n    return other is BoxedChar && c == other.c\n  }\n\n  override fun
hashCode(): Int {\n    return c\n  }\n\n  override fun toString(): String {\n    return
js("this.c").unsafeCast<Char>().toString()\n  }\n\n  override fun compareTo(other: Int): Int {\n    return
js("this.c - other").unsafeCast<Int>()\n  }\n\n  @JsName("valueOf")\n  public fun valueOf(): Int {\n
return c\n  }\n}\n\n@kotlin.internal.InlineOnly\ninternal inline fun <T> concat(args: Array<T>): T {\n  val typed
= js("Array")(args.size)\n  for (i in args.indices) {\n    val arr = args[i]\n    if (arr !is Array<*>) {\n
typed[i] = js("[]").slice.call(arr)\n    } else {\n    typed[i] = arr\n    }\n  }\n  return
js("[]").concat.apply(js("[]"), typed);\n}\n\n/** Concat regular Array's and TypedArray's into an Array.\n
*/\n\n@PublishedApi\n@JsName("arrayConcat")\n@Suppress("UNUSED_PARAMETER")\ninternal fun <T>
arrayConcat(a: T, b: T): T {\n  return concat(js("arguments"))\n}\n\n/** Concat primitive arrays. Main use:
prepare vararg arguments.\n
* For compatibility with 1.1.0 the arguments may be a mixture of Array's and
TypedArray's.\n
* If the first argument is TypedArray (Byte-, Short-, Char-, Int-, Float-, and DoubleArray)
returns a TypedArray, otherwise an Array.\n
* If the first argument has the $type$ property (Boolean-, Char-, and
LongArray) copy its value to result.$type$.\n
* If the first argument is a regular Array without the $type$ property
default to arrayConcat.\n
*/\n\n@PublishedApi\n@JsName("primitiveArrayConcat")\n@Suppress("UNUSED_PARAMETER")\ninternal
fun <T> primitiveArrayConcat(a: T, b: T): T {\n  val args: Array<T> = js("arguments")\n  if (a is Array<*> &&
a.asDynamic().$type$ === undefined) {\n    return concat(args)\n  } else {\n    var size = 0\n    for (i in
args.indices) {\n      size += args[i].asDynamic().length as Int\n    }\n    val result = js("new
a.constructor(size)")\n    kotlin.copyArrayType(a, result)\n    size = 0\n    for (i in args.indices) {\n      val
arr = args[i].asDynamic()\n      for (j in 0 until arr.length) {\n        result[size++] = arr[j]\n      }\n    }\n
return result\n  }\n}\n\n@JsName("booleanArrayOf")\ninternal fun booleanArrayOf() =
withType("BooleanArray", js("[]").slice.call(arguments))\n\n@JsName("charArrayOf") // The arguments have
to be slice'd here because of Rhino (see KT-16974)\ninternal fun charArrayOf() = withType("CharArray", js("new
Uint16Array([]).slice.call(arguments)))\n\n@JsName("longArrayOf")\ninternal fun longArrayOf() =

```

```

withType("LongArray",
js("[]slice.call(arguments)")\n\n@JsName("withType")\n@kotlin.internal.InlineOnly\ninternal inline fun
withType(type: String, array: dynamic): dynamic {\n    array.`$type$` = type\n    return array\n}\n", /*\n * Copyright
2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed
by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.js\n\n/**\n *
Function corresponding to JavaScript's `typeof` operator\n
*/\n@kotlin.internal.InlineOnly\n@Suppress("UNUSED_PARAMETER")\npublic inline fun jsTypeOf(a: Any?):
String = js("typeof a")\n", /*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\n@file:Suppress("UNUSED_PARAMETER",
"NOTHING_TO_INLINE")\n\npackage kotlin\n\n/**\n * Returns an empty array of the specified type [T].\n
*/\n\npublic inline fun <T> emptyArray(): Array<T> = js("[]")\n\n@library\npublic fun <T> arrayOf(vararg
elements: T): Array<T> = definedExternally\n\n@library\npublic fun doubleArrayOf(vararg elements: Double):
DoubleArray = definedExternally\n\n@library\npublic fun floatArrayOf(vararg elements: Float): FloatArray =
definedExternally\n\n@library\npublic fun longArrayOf(vararg elements: Long): LongArray =
definedExternally\n\n@library\npublic fun intArrayOf(vararg elements: Int): IntArray =
definedExternally\n\n@library\npublic fun charArrayOf(vararg elements: Char): CharArray =
definedExternally\n\n@library\npublic fun shortArrayOf(vararg elements: Short): ShortArray =
definedExternally\n\n@library\npublic fun byteArrayOf(vararg elements: Byte): ByteArray =
definedExternally\n\n@library\npublic fun booleanArrayOf(vararg elements: Boolean): BooleanArray =
definedExternally\n\n/**\n * Creates a new instance of the [Lazy] that uses the specified initialization function
[initializer].\n */\n\npublic actual fun <T> lazy(initializer: () -> T): Lazy<T> = UnsafeLazyImpl(initializer)\n\n/**\n *
Creates a new instance of the [Lazy] that uses the specified initialization function [initializer].\n */\n\n * The [mode]
parameter is ignored.\n */\n\npublic actual fun <T> lazy(mode: LazyThreadSafetyMode, initializer: () -> T): Lazy<T> =
UnsafeLazyImpl(initializer)\n\n/**\n * Creates a new instance of the [Lazy] that uses the specified initialization
function [initializer].\n */\n\n * The [lock] parameter is ignored.\n */\n\npublic actual fun <T> lazy(lock: Any?,
initializer: () -> T): Lazy<T> = UnsafeLazyImpl(initializer)\n\n\ninternal fun fillFrom(src: dynamic, dst: dynamic):
dynamic {\n    val srcLen: Int = src.length\n    val dstLen: Int = dst.length\n    var index: Int = 0\n    while (index <
srcLen && index < dstLen) dst[index] = src[index++]\n    return dst\n}\n\n\ninternal fun arrayCopyResize(source:
dynamic, newSize: Int, defaultValue: Any?): dynamic {\n    val result = source.slice(0, newSize)\n
copyArrayType(source, result)\n    var index: Int = source.length\n    if (newSize > index) {\n        result.length =
newSize\n        while (index < newSize) result[index++] = defaultValue\n    }\n    return result\n}\n\n\ninternal fun
<T> arrayPlusCollection(array: dynamic, collection: Collection<T>): dynamic {\n    val result = array.slice()\n
result.length += collection.size\n    copyArrayType(array, result)\n    var index: Int = array.length\n    for (element in
collection) result[index++] = element\n    return result\n}\n\n\ninternal fun <T> fillFromCollection(dst: dynamic,
startIndex: Int, collection: Collection<T>): dynamic {\n    var index = startIndex\n    for (element in collection)
dst[index++] = element\n    return dst\n}\n\n\ninternal inline fun copyArrayType(from: dynamic, to: dynamic) {\n    if
(from.`$type$` !== undefined) {\n        to.`$type$` = from.`$type$`\n    }\n}\n\n\ninternal inline fun jsIsType(obj:
dynamic, jsClass: dynamic) = js("Kotlin").isType(obj, jsClass)", /*\n * Copyright 2010-2021 JetBrains s.r.o. and
Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin\n\n/**\n * Creates a Char with the specified
[code].\n */\n\n * @sample samples.text.Chars.charFromCode\n
*/\n\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
c actual inline fun Char(code: UShort): Char {\n    return code.toInt().toChar()\n}\n", /*\n * Copyright 2010-2018
JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.coroutines\n\nimport
kotlin.coroutines.intrinsics.COROUTINE_SUSPENDED\n\n@SinceKotlin("1.3")\n@JsName("CoroutineImpl")\n
ninternal abstract class CoroutineImpl(private val resultContinuation: Continuation<Any?>) : Continuation<Any?>

```

```

{\n    protected var state = 0\n    protected var exceptionState = 0\n    protected var result: Any? = null\n    protected
var exception: Throwable? = null\n    protected var finallyPath: Array<Int>? = null\n\n    public override val context:
CoroutineContext = resultContinuation.context\n\n    private var intercepted_: Continuation<Any?>? = null\n\n
public fun intercepted(): Continuation<Any?> =\n        intercepted_\n            ?:\n    (context[ContinuationInterceptor]?.interceptContinuation(this) ?: this)\n        .also { intercepted_ = it }\n\n
override fun resumeWith(result: Result<Any?>) {\n        var current = this\n        var currentResult: Any? =
result.getOrNull()\n        var currentException: Throwable? = result.exceptionOrNull()\n        // This loop unrolls
recursion in current.resumeWith(param) to make saner and shorter stack traces on resume\n        while (true) {\n
            with(current) {\n                val completion = resultContinuation\n                // Set result and exception fields in
the current continuation\n                if (currentException == null) {\n                    this.result = currentResult\n
                } else {\n                    state = exceptionState\n                    exception = currentException\n                }\n
            }\n            try {\n                val outcome = doResume()\n                if (outcome === COROUTINE_SUSPENDED)\n            }\n            return\n                currentResult = outcome\n                currentException = null\n            } catch (exception:
dynamic) { // Catch all exceptions\n                currentResult = null\n                currentException =
exception.unsafeCast<Throwable>()\n            }\n            releaseIntercepted() // this state machine instance is
terminating\n\n            if (completion is CoroutineImpl) {\n                // unrolling recursion via loop\n                current = completion\n            } else {\n                // top-level completion reached -- invoke and return\n                currentException?.let {\n                    completion.resumeWithException(it)\n                } ?:\n                completion.resume(currentResult)\n                return\n            }\n        }\n        }\n        }\n        }\n        }\n        private fun
releaseIntercepted() {\n            val intercepted = intercepted_\n            if (intercepted != null && intercepted !== this) {\n
                context[ContinuationInterceptor]!!.releaseInterceptedContinuation(intercepted)\n            }\n            this.intercepted_
= CompletedContinuation // just in case\n        }\n        protected abstract fun doResume(): Any?\n    }\n\n    internal object
CompletedContinuation : Continuation<Any?> {\n        override val context: CoroutineContext\n            get() =
error("This continuation is already complete")\n\n        override fun resumeWith(result: Result<Any?>) {\n
            error("This continuation is already complete")\n        }\n\n        override fun toString(): String = "This continuation is
already complete"\n    }\n\n    /*\n     * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n     * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n     */\n\n    @file:Suppress("UNCHECKED_CAST",
"RedundantVisibilityModifier")\n\n    package kotlin\n\n    import kotlin.contracts.*\n    import
kotlin.internal.InlineOnly\n    import kotlin.jvm.JvmField\n    import kotlin.jvm.JvmInline\n    import
kotlin.jvm.JvmName\n\n    /**\n     * A discriminated union that encapsulates a successful outcome with a value of type
[T]\n     * or a failure with an arbitrary [Throwable] exception.\n     */\n\n    @SinceKotlin("1.3")\n    @JvmInline\n    public
value class Result<out T> @PublishedApi internal constructor(\n        @PublishedApi\n        internal val value: Any?\n    ) :
Serializable {\n        // discovery\n        /**\n         * Returns `true` if this instance represents a successful outcome.\n         *
In this case [isFailure] returns `false`.\n         */\n        public val isSuccess: Boolean\n            get() = value !is Failure\n        /**\n         * Returns `true` if this instance represents a failed outcome.\n         *
In this case [isSuccess] returns `false`.\n         */\n        public val isFailure: Boolean\n            get() = value is Failure\n\n        // value & exception retrieval\n        /**\n         * Returns the
encapsulated value if this instance represents [success][Result.isSuccess] or `null` if it is
[failure][Result.isFailure].\n         */\n        * This function is a shorthand for `getOrNull` (see [getOrNull]) or\n
        * `fold(onSuccess = { it }, onFailure = { null })` (see [fold]).\n        */\n        @InlineOnly\n        public inline fun
getOrNull(): T? =\n            when {\n                isFailure -> null\n                else -> value as T\n            }\n        /**\n         *
Returns the encapsulated [Throwable] exception if this instance represents [failure][isFailure] or `null` if it is
[success][isSuccess].\n         */\n        * This function is a shorthand for `fold(onSuccess = { null }, onFailure = { it })`
(see [fold]).\n        */\n        public fun exceptionOrNull(): Throwable? =\n            when (value) {\n                is Failure ->
value.exception\n                else -> null\n            }\n        /**\n         * Returns a string `Success(v)` if this instance represents
[success][Result.isSuccess]\n         * where `v` is a string representation of the value or a string `Failure(x)` if\n         * it
is [failure][isFailure] where `x` is a string representation of the exception.\n         */\n        public override fun toString():
String =\n            when (value) {\n                is Failure -> value.toString() // "Failure($exception)"\n                else ->

```

```

"Success($value)"\n    }\n    // companion with constructors\n    /**\n     * Companion object for [Result]
class that contains its constructor functions\n     * [success] and [failure].\n     */\n    public companion object {\n
/**\n     * Returns an instance that encapsulates the given [value] as successful value.\n     */\n
@Suppress("INAPPLICABLE_JVM_NAME")\n     @InlineOnly\n     @JvmName("success")\n     public
inline fun <T> success(value: T): Result<T> =\n         Result(value)\n    /**\n     * Returns an instance that
encapsulates the given [Throwable] [exception] as failure.\n     */\n
@Suppress("INAPPLICABLE_JVM_NAME")\n     @InlineOnly\n     @JvmName("failure")\n     public
inline fun <T> failure(exception: Throwable): Result<T> =\n         Result(createFailure(exception))\n    }\n
internal class Failure(\n        @JvmField\n        val exception: Throwable\n    ): Serializable {\n        override fun
equals(other: Any?): Boolean = other is Failure && exception == other.exception\n        override fun hashCode():
Int = exception.hashCode()\n        override fun toString(): String = "Failure($exception)"\n    }\n\n/**\n     *
Creates an instance of internal marker [Result.Failure] class to\n     * make sure that this class is not exposed in ABI.\n
*/\n    @PublishedApi\n    @SinceKotlin("1.3")\n    internal fun createFailure(exception: Throwable): Any =\n
Result.Failure(exception)\n\n/**\n     * Throws exception if the result is failure. This internal function minimizes\n
*/\n    inlined bytecode for [getOrThrow] and makes sure that in the future we can\n     * add some exception-augmenting
logic here (if needed).\n     */\n    @PublishedApi\n    @SinceKotlin("1.3")\n    internal fun Result<*>.throwOnFailure() {\n
if (value is Result.Failure) throw value.exception\n    }\n\n/**\n     * Calls the specified function [block] and returns its
encapsulated result if invocation was successful,\n     * catching any [Throwable] exception that was thrown from the
[block] function execution and encapsulating it as a failure.\n     */\n    @InlineOnly\n    @SinceKotlin("1.3")\n    public
inline fun <R> runCatching(block: () -> R): Result<R> {\n        return try {\n            Result.success(block())\n        } catch
(e: Throwable) {\n            Result.failure(e)\n        }\n    }\n\n/**\n     * Calls the specified function [block] with `this` value as
its receiver and returns its encapsulated result if invocation was successful,\n     * catching any [Throwable] exception
that was thrown from the [block] function execution and encapsulating it as a failure.\n
*/\n    @InlineOnly\n    @SinceKotlin("1.3")\n    public inline fun <T, R> T.runCatching(block: T.() -> R): Result<R> {\n
return try {\n        Result.success(block())\n    } catch (e: Throwable) {\n        Result.failure(e)\n    }\n}\n\n// --
extensions ---\n\n/**\n     * Returns the encapsulated value if this instance represents [success][Result.isSuccess] or
throws the encapsulated [Throwable] exception\n     * if it is [failure][Result.isFailure].\n     */\n     * This function is a
shorthand for `getOrNull { throw it }` (see [getOrNull]).\n     */\n    @InlineOnly\n    @SinceKotlin("1.3")\n    public inline
fun <T> Result<T>.getOrThrow(): T {\n        throwOnFailure()\n        return value as T\n    }\n\n/**\n     * Returns the
encapsulated value if this instance represents [success][Result.isSuccess] or the\n     * result of [onFailure] function for
the encapsulated [Throwable] exception if it is [failure][Result.isFailure].\n     */\n     * Note, that this function rethrows
any [Throwable] exception thrown by [onFailure] function.\n     */\n     * This function is a shorthand for `fold(onSuccess
= { it }, onFailure = onFailure)` (see [fold]).\n     */\n    @InlineOnly\n    @SinceKotlin("1.3")\n    public inline fun <R, T :
R> Result<T>.getOrNull(onFailure: (exception: Throwable) -> R): R {\n        contract {\n            callsInPlace(onFailure,
InvocationKind.AT_MOST_ONCE)\n        }\n        return when (val exception = exceptionOrNull()) {\n            null ->
value as T\n            else -> onFailure(exception)\n        }\n    }\n\n/**\n     * Returns the encapsulated value if this instance
represents [success][Result.isSuccess] or the\n     * [defaultValue] if it is [failure][Result.isFailure].\n     */\n     * This
function is a shorthand for `getOrNull { defaultValue }` (see [getOrNull]).\n
*/\n    @InlineOnly\n    @SinceKotlin("1.3")\n    public inline fun <R, T : R> Result<T>.getOrDefault(defaultValue: R):
R {\n        if (isFailure) return defaultValue\n        return value as T\n    }\n\n/**\n     * Returns the result of [onSuccess] for the
encapsulated value if this instance represents [success][Result.isSuccess]\n     * or the result of [onFailure] function for
the encapsulated [Throwable] exception if it is [failure][Result.isFailure].\n     */\n     * Note, that this function rethrows
any [Throwable] exception thrown by [onSuccess] or by [onFailure] function.\n
*/\n    @InlineOnly\n    @SinceKotlin("1.3")\n    public inline fun <R, T> Result<T>.fold(\n        onSuccess: (value: T) ->
R,\n        onFailure: (exception: Throwable) -> R\n    ): R {\n        contract {\n            callsInPlace(onSuccess,
InvocationKind.AT_MOST_ONCE)\n            callsInPlace(onFailure, InvocationKind.AT_MOST_ONCE)\n        }\n        return when (val exception = exceptionOrNull()) {\n            null -> onSuccess(value as T)\n            else ->
onFailure(exception)\n        }\n    }\n\n// transformation\n\n/**\n     * Returns the encapsulated result of the given

```

[transform] function applied to the encapsulated value\n \* if this instance represents [success][Result.isSuccess] or the\n \* original encapsulated [Throwable] exception if it is [failure][Result.isFailure].\n \* Note, that this function rethrows any [Throwable] exception thrown by [transform] function.\n \* See [mapCatching] for an alternative that encapsulates exceptions.\n \*  
`@InlineOnly\n@SinceKotlin("1.3")\npublic inline fun <R, T>  
Result<T>.map(transform: (value: T) -> R): Result<R> {\n contract {\n callsInPlace(transform,  
InvocationKind.AT_MOST_ONCE)\n }\n return when {\n isSuccess -> Result.success(transform(value as  
T))\n else -> Result(value)\n }\n}\n\n/**\n * Returns the encapsulated result of the given [transform] function  
applied to the encapsulated value\n * if this instance represents [success][Result.isSuccess] or the\n * original  
encapsulated [Throwable] exception if it is [failure][Result.isFailure].\n * This function catches any [Throwable]  
exception thrown by [transform] function and encapsulates it as a failure.\n * See [map] for an alternative that  
rethrows exceptions from `transform` function.\n *  
@InlineOnly\n@SinceKotlin("1.3")\npublic inline fun <R,  
T> Result<T>.mapCatching(transform: (value: T) -> R): Result<R> {\n return when {\n isSuccess ->  
runCatching { transform(value as T) }\n else -> Result(value)\n }\n}\n\n/**\n * Returns the encapsulated  
result of the given [transform] function applied to the encapsulated [Throwable] exception\n * if this instance  
represents [failure][Result.isFailure] or the\n * original encapsulated value if it is [success][Result.isSuccess].\n * Note, that this function rethrows any [Throwable] exception thrown by [transform] function.\n * See  
[recoverCatching] for an alternative that encapsulates exceptions.\n  
@InlineOnly\n@SinceKotlin("1.3")\npublic inline fun <R, T : R> Result<T>.recover(transform: (exception:  
Throwable) -> R): Result<R> {\n contract {\n callsInPlace(transform, InvocationKind.AT_MOST_ONCE)\n  
}\n return when (val exception = exceptionOrNull()) {\n null -> this\n else ->  
Result.success(transform(exception))\n }\n}\n\n/**\n * Returns the encapsulated result of the given [transform]  
function applied to the encapsulated [Throwable] exception\n * if this instance represents [failure][Result.isFailure]  
or the\n * original encapsulated value if it is [success][Result.isSuccess].\n * This function catches any  
[Throwable] exception thrown by [transform] function and encapsulates it as a failure.\n * See [recover] for an  
alternative that rethrows exceptions.\n *  
@InlineOnly\n@SinceKotlin("1.3")\npublic inline fun <R, T : R>  
Result<T>.recoverCatching(transform: (exception: Throwable) -> R): Result<R> {\n return when (val exception =  
exceptionOrNull()) {\n null -> this\n else -> runCatching { transform(exception) }\n }\n}\n\n/**\n * Performs the given [action] on the encapsulated [Throwable] exception if  
this instance represents [failure][Result.isFailure].\n * Returns the original `Result` unchanged.\n  
@InlineOnly\n@SinceKotlin("1.3")\npublic inline fun <T> Result<T>.onFailure(action: (exception:  
Throwable) -> Unit): Result<T> {\n contract {\n callsInPlace(action, InvocationKind.AT_MOST_ONCE)\n  
}\n exceptionOrNull()?.let { action(it) }\n return this\n}\n\n/**\n * Performs the given [action] on the  
encapsulated value if this instance represents [success][Result.isSuccess].\n * Returns the original `Result`  
unchanged.\n *  
@InlineOnly\n@SinceKotlin("1.3")\npublic inline fun <T> Result<T>.onSuccess(action: (value:  
T) -> Unit): Result<T> {\n contract {\n callsInPlace(action, InvocationKind.AT_MOST_ONCE)\n }\n if  
(isSuccess) action(value as T)\n return this\n}\n\n/**\n * Copyright 2010-2020 JetBrains  
s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0  
license that can be found in the license/LICENSE.txt file.\n *  
package kotlin.coroutines\n\nimport  
kotlin.contracts.*\nimport kotlin.coroutines.intrinsics.*\nimport kotlin.internal.InlineOnly\n\n/**\n * Interface  
representing a continuation after a suspension point that returns a value of type `T`.\n  
@SinceKotlin("1.3")\npublic interface Continuation<in T> {\n /**\n * The context of the coroutine that  
corresponds to this continuation.\n *  
public val context: CoroutineContext\n\n /**\n * Resumes the  
execution of the corresponding coroutine passing a successful or failed [result] as the\n * return value of the last  
suspension point.\n *  
public fun resumeWith(result: Result<T>)\n}\n\n/**\n * Classes and interfaces marked  
with this annotation are restricted when used as receivers for extension\n * `suspend` functions. These `suspend`  
extensions can only invoke other member or extension `suspend` functions on this particular\n * receiver and are  
restricted from calling arbitrary suspension functions.\n  
@SinceKotlin("1.3")\n@Target(AnnotationTarget.CLASS)\n@Retention(AnnotationRetention.BINARY)\n`

blic annotation class RestrictsSuspension\n\n\*\*\n \* Resumes the execution of the corresponding coroutine passing [value] as the return value of the last suspension point.\n \*^\n@SinceKotlin("1.3")\n@InlineOnly\npublic inline fun <T> Continuation<T>.resume(value: T): Unit =\n resumeWith(Result.success(value))\n\n\*\*\n \* Resumes the execution of the corresponding coroutine so that the [exception] is re-thrown right after the\n \* last suspension point.\n \*^\n@SinceKotlin("1.3")\n@InlineOnly\npublic inline fun <T> Continuation<T>.resumeWithException(exception: Throwable): Unit =\n resumeWith(Result.failure(exception))\n\n\*\*\n \* Creates a [Continuation] instance with the given [context] and implementation of [resumeWith] method.\n \*^\n@SinceKotlin("1.3")\n@InlineOnly\npublic inline fun <T> Continuation(\n context: CoroutineContext,\n crossinline resumeWith: (Result<T>) -> Unit\n): Continuation<T> =\n object : Continuation<T> {\n override val context: CoroutineContext\n get() = context\n\n override fun resumeWith(result: Result<T>) =\n resumeWith(result)\n }\n\n\*\*\n \* Creates a coroutine without a receiver and with result type [T].\n \* This function creates a new, fresh instance of suspendable computation every time it is invoked.\n \* To start executing the created coroutine, invoke `resume(Unit)` on the returned [Continuation] instance.\n \* The [completion] continuation is invoked when the coroutine completes with a result or an exception.\n \* Subsequent invocation of any resume function on the resulting continuation will produce an [IllegalStateException].\n \*^\n@SinceKotlin("1.3")\n@Suppress("UNCHECKED\_CAST")\npublic fun <T> (suspend () -> T).createCoroutine(\n completion: Continuation<T>)\n): Continuation<Unit> =\n SafeContinuation(createCoroutineUnintercepted(completion).intercepted(), COROUTINE\_SUSPENDED)\n\n\*\*\n \* Creates a coroutine with receiver type [R] and result type [T].\n \* This function creates a new, fresh instance of suspendable computation every time it is invoked.\n \* To start executing the created coroutine, invoke `resume(Unit)` on the returned [Continuation] instance.\n \* The [completion] continuation is invoked when the coroutine completes with a result or an exception.\n \* Subsequent invocation of any resume function on the resulting continuation will produce an [IllegalStateException].\n \*^\n@SinceKotlin("1.3")\n@Suppress("UNCHECKED\_CAST")\npublic fun <R, T> (suspend R.() -> T).createCoroutine(\n receiver: R,\n completion: Continuation<T>)\n): Continuation<Unit> =\n SafeContinuation(createCoroutineUnintercepted(receiver, completion).intercepted(), COROUTINE\_SUSPENDED)\n\n\*\*\n \* Starts a coroutine without a receiver and with result type [T].\n \* This function creates and starts a new, fresh instance of suspendable computation every time it is invoked.\n \* The [completion] continuation is invoked when the coroutine completes with a result or an exception.\n \*^\n@SinceKotlin("1.3")\n@Suppress("UNCHECKED\_CAST")\npublic fun <T> (suspend () -> T).startCoroutine(\n completion: Continuation<T>)\n) {\n createCoroutineUnintercepted(completion).intercepted().resume(Unit)\n}\n\n\*\*\n \* Starts a coroutine with receiver type [R] and result type [T].\n \* This function creates and starts a new, fresh instance of suspendable computation every time it is invoked.\n \* The [completion] continuation is invoked when the coroutine completes with a result or an exception.\n \*^\n@SinceKotlin("1.3")\n@Suppress("UNCHECKED\_CAST")\npublic fun <R, T> (suspend R.() -> T).startCoroutine(\n receiver: R,\n completion: Continuation<T>)\n) {\n createCoroutineUnintercepted(receiver, completion).intercepted().resume(Unit)\n}\n\n\*\*\n \* Obtains the current continuation instance inside suspend functions and suspends\n \* the currently running coroutine.\n \* In this function both [Continuation.resume] and [Continuation.resumeWithException] can be used either synchronously in\n \* the same stack-frame where the suspension function is run or asynchronously later in the same thread or\n \* from a different thread of execution. Subsequent invocation of any resume function will produce an [IllegalStateException].\n \*^\n@SinceKotlin("1.3")\n@InlineOnly\npublic suspend inline fun <T> suspendCoroutine(crossinline block: (Continuation<T>) -> Unit): T {\n contract { callsInPlace(block, InvocationKind.EXACTLY\_ONCE) }\n return suspendCoroutineUninterceptedOrReturn { c: Continuation<T> ->\n val safe = SafeContinuation(c.intercepted())\n block(safe)\n safe.getOrThrow()\n }\n}\n\n\*\*\n \* Returns the context of the current coroutine.\n \*^\n@SinceKotlin("1.3")\n@Suppress("WRONG\_MODIFIER\_TARGET")\n@InlineOnly\npublic suspend inline val coroutineContext: CoroutineContext\n get() {\n throw NotImplementedError("Implemented as

```

intrinsic")\n } \n", /*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.coroutines.intrinsics\n\nimport kotlin.coroutines.*\nimport kotlin.internal.InlineOnly\n\n/**\n * Starts an unintercepted coroutine without a receiver and with result type [T] and executes it until its first\n * suspension.\n * Returns the result of the coroutine or throws its exception if it does not suspend or\n * [COROUTINE_SUSPENDED] if it suspends.\n * In the latter case, the [completion] continuation is invoked when\n * the coroutine completes with a result or an exception.\n * The coroutine is started directly in the invoker's thread\n * without going through the [ContinuationInterceptor] that might\n * be present in the completion's\n * [CoroutineContext]. It is the invoker's responsibility to ensure that a proper invocation\n * context is established.\n * This function is designed to be used from inside of [suspendCoroutineUninterceptedOrReturn] to resume the\n * execution of the suspended\n * coroutine using a reference to the suspending function.\n */\n@SinceKotlin("1.3")\n@InlineOnly\npublic actual inline fun <T> (suspend () ->\n T).startCoroutineUninterceptedOrReturn(\n completion: Continuation<T>)\n: Any? =\n this.asDynamic()(completion, false)\n\n/**\n * Starts an unintercepted coroutine with receiver type [R] and result\n * type [T] and executes it until its first suspension.\n * Returns the result of the coroutine or throws its exception if it\n * does not suspend or [COROUTINE_SUSPENDED] if it suspends.\n * In the latter case, the [completion]\n * continuation is invoked when the coroutine completes with a result or an exception.\n * The coroutine is started\n * directly in the invoker's thread without going through the [ContinuationInterceptor] that might\n * be present in the\n * completion's [CoroutineContext]. It is the invoker's responsibility to ensure that a proper invocation\n * context is\n * established.\n * This function is designed to be used from inside of [suspendCoroutineUninterceptedOrReturn]\n * to resume the execution of the suspended\n * coroutine using a reference to the suspending function.\n */\n@SinceKotlin("1.3")\n@InlineOnly\npublic actual inline fun <R, T> (suspend R.() ->\n T).startCoroutineUninterceptedOrReturn(\n receiver: R,\n completion: Continuation<T>)\n: Any? =\n this.asDynamic()(receiver, completion, false)\n\n@InlineOnly\ninternal actual inline fun <R, P, T> (suspend R.(P) ->\n T).startCoroutineUninterceptedOrReturn(\n receiver: R,\n param: P,\n completion: Continuation<T>)\n: Any? = this.asDynamic()(receiver, param, completion, false)\n\n/**\n * Creates unintercepted coroutine without\n * receiver and with result type [T].\n * This function creates a new, fresh instance of suspendable computation every\n * time it is invoked.\n * To start executing the created coroutine, invoke `resume(Unit)` on the returned\n * [Continuation] instance.\n * The [completion] continuation is invoked when coroutine completes with result or\n * exception.\n * This function returns unintercepted continuation.\n * Invocation of `resume(Unit)` starts coroutine\n * immediately in the invoker's call stack without going through the\n * [ContinuationInterceptor] that might be present\n * in the completion's [CoroutineContext].\n * It is the invoker's responsibility to ensure that a proper invocation\n * context is established.\n * Note that [completion] of this function may get invoked in an arbitrary context.\n * [Continuation.intercepted] can be used to acquire the intercepted continuation.\n * Invocation of `resume(Unit)` on\n * intercepted continuation guarantees that execution of\n * both the coroutine and [completion] happens in the\n * invocation context established by\n * [ContinuationInterceptor].\n * Repeated invocation of any resume function\n * on the resulting continuation corrupts the\n * state machine of the coroutine and may result in arbitrary behaviour or\n * exception.\n */\n@SinceKotlin("1.3")\npublic actual fun <T> (suspend () -> T).createCoroutineUnintercepted(\n completion: Continuation<T>)\n: Continuation<Unit> =\n // Kotlin/JS suspend lambdas have an extra parameter\n `suspended`\n if (this.asDynamic().length == 2) {\n // When `suspended` is true the continuation is created,\n but not executed\n this.asDynamic()(completion, true)\n } else {\n createCoroutineFromSuspendFunction(completion) {\n this.asDynamic()(completion)\n }\n }\n\n/**\n * Creates unintercepted coroutine with receiver type [R] and result type [T].\n * This function creates a new, fresh\n * instance of suspendable computation every time it is invoked.\n * To start executing the created coroutine,\n * invoke `resume(Unit)` on the returned [Continuation] instance.\n * The [completion] continuation is invoked when\n * coroutine completes with result or exception.\n * This function returns unintercepted continuation.\n * Invocation of `resume(Unit)` starts coroutine immediately in the invoker's call stack without going through the\n * [ContinuationInterceptor] that might be present in the completion's [CoroutineContext].\n * It is the invoker's

```

responsibility to ensure that a proper invocation context is established.

\* Note that [completion] of this function may get invoked in an arbitrary context.

\* [Continuation.intercepted] can be used to acquire the intercepted continuation.

\* Invocation of `resume(Unit)` on intercepted continuation guarantees that execution of both the coroutine and [completion] happens in the invocation context established by [ContinuationInterceptor].

\* Repeated invocation of any resume function on the resulting continuation corrupts the state machine of the coroutine and may result in arbitrary behaviour or exception.

```

@SinceKotlin("1.3")
public actual fun <R, T>
(suspend R.() -> T).createCoroutineUnintercepted(
    receiver: R,
    completion: Continuation<T>):
Continuation<Unit> =
    // Kotlin/JS suspend lambdas have an extra parameter `suspended`
    if
    (this.asDynamic().length == 3) {
        // When `suspended` is true the continuation is created, but not executed
        this.asDynamic()(receiver, completion, true)
    } else {
        createCoroutineFromSuspendFunction(completion)
    }
    this.asDynamic()(receiver, completion)
}

// Intercepts this continuation with
[ContinuationInterceptor].
// This function shall be used on the immediate result of
[createCoroutineUnintercepted] or [suspendCoroutineUninterceptedOrReturn],
// in which case it checks for
[ContinuationInterceptor] in the continuation's [context][Continuation.context],
// invokes
[ContinuationInterceptor.interceptContinuation], caches and returns the result.
// If this function is invoked on
other [Continuation] instances it returns `this` continuation unchanged.
@SinceKotlin("1.3")
public actual
fun <T> Continuation<T>.intercepted(): Continuation<T> =
    (this as? CoroutineImpl)?.intercepted() ?:
    this
private inline fun <T> createCoroutineFromSuspendFunction(
    completion: Continuation<T>,
    crossinline block: () -> Any?): Continuation<Unit> {
    @Suppress("UNCHECKED_CAST")
    return object
    : CoroutineImpl(completion as Continuation<Any?>) {
        override fun doResume(): Any? {
            exception?.let { throw it }
            return block()
        }
    }
}

// Copyright 2010-2018 JetBrains s.r.o.
and Kotlin Programming Language contributors.
// Use of this source code is governed by the Apache 2.0 license
that can be found in the license/LICENSE.txt file.
package kotlin.js
// Mirrors signature from JS IR
BE
// Used for
js.translator/testData/box/number/mulInt32.kt
@library
@JsName("imulEmulated")
@Suppress("UNUSED_PARAMETER")
internal fun imul(x: Int, y: Int): Int =
    definedExternally
@Suppress("NOTHING_TO_INLINE")
internal inline fun isArrayish(o: dynamic) =
    js("Kotlin").isArrayish(o)
// Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
contributors.
// Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.
package kotlin
// NOTE: Do not author your exceptions as they are written in
this file, instead use this template:
public open class MyException : Exception {
    constructor() : super()
    constructor(message: String?) : super(message)
    constructor(message: String?, cause: Throwable?) :
    super(message, cause)
    constructor(cause: Throwable?) : super(cause)
}
// TODO: remove primary
constructors, make all secondary
@Suppress("USELESS_ELVIS_RIGHT_IS_NULL")
public
actual open class Error actual constructor(message: String?, cause: Throwable?) : Throwable(message, cause ?: null)
{
    actual constructor() : this(null, null)
    actual constructor(message: String?) : this(message, null)
    actual
    constructor(cause: Throwable?) : this(undefined,
    cause)
}
@Suppress("USELESS_ELVIS_RIGHT_IS_NULL")
public
actual open class Exception actual
constructor(message: String?, cause: Throwable?) : Throwable(message, cause ?: null) {
    actual constructor() :
    this(null, null)
    actual constructor(message: String?) : this(message, null)
    actual constructor(cause:
    Throwable?) : this(undefined, cause)
}
public
actual open class RuntimeException actual constructor(message:
String?, cause: Throwable?) : Exception(message, cause) {
    actual constructor() : this(null, null)
    actual
    constructor(message: String?) : this(message, null)
    actual constructor(cause: Throwable?) : this(undefined,
    cause)
}
public
actual open class IllegalArgumentException actual constructor(message: String?, cause:
Throwable?) : RuntimeException(message, cause) {
    actual constructor() : this(null, null)
    actual
    constructor(message: String?) : this(message, null)
    actual constructor(cause: Throwable?) : this(undefined,
    cause)
}
public
actual open class IllegalStateException actual constructor(message: String?, cause: Throwable?) :
RuntimeException(message, cause) {
    actual constructor() : this(null, null)
    actual constructor(message:

```

```

String?): this(message, null)\n  actual constructor(cause: Throwable?): this(undefi
ned, cause)\n}\n\npublic actual
open class IndexOutOfBoundsException actual constructor(message: String?): Runtime
Exception(message) {\n
actual constructor(): this(null)\n}\n\npublic actual open class ConcurrentModifi
cationException actual
constructor(message: String?, cause: Throwable?): RuntimeException(message, cause)
{\n  actual constructor():
this(null, null)\n  actual constructor(message: String?): this(message, null)\n
  actual constructor(cause:
Throwable?): this(undefi
ned, cause)\n}\n\npublic actual open class UnsupportedOperationException actual
constructor(message: String?, cause: Throwable?): RuntimeException(message, cause)
{\n  actual constructor():
this(null, null)\n  actual constructor(message: String?): this(message, null)\n
  actual constructor(cause:
Throwable?): this(undefi
ned, cause)\n}\n\npublic actual open class NumberFormatException actual
constructor(message: String?): IllegalArgument
Exception(message) {\n  actual constructor():
this(null)\n}\n\npublic actual open class NullPointerException actual constructor
(message: String?):
RuntimeException(message) {\n  actual constructor(): this(null)\n}\n\npublic actual
open class
ClassCastException actual constructor(message: String?): RuntimeException(message)
{\n  actual constructor():
this(null)\n}\n\npublic actual open class AssertionError\n@SinceKotlin("1.4")\ncon
structor(message: String?,
cause: Throwable?): Error(message, cause) {\n  actual constructor(): this(null)\n
  constructor(message: String?):
this(message, null)\n  actual constructor(message: Any?): this(message.toString(),
message as?
Throwable)\n}\n\npublic actual open class NoSuchElementException actual constructor
(message: String?):
RuntimeException(message) {\n  actual constructor(): this(null)\n}\n\n@SinceKotl
in("1.3")\n\npublic actual open
class ArithmeticException actual constructor(message: String?): RuntimeException
(message) {\n  actual
constructor(): this(null)\n}\n\npublic actual open class NoWhenBranchMatchedEx
ception actual
constructor(message: String?, cause: Throwable?): RuntimeException(message, cause)
{\n  actual constructor():
this(null, null)\n  actual constructor(message: String?): this(message, null)\n
  actual constructor(cause:
Throwable?): this(undefi
ned, cause)\n}\n\npublic actual open class UninitializedPropertyAccessExceptio
n actual
constructor(message: String?, cause: Throwable?): RuntimeException(message, cause)
{\n  actual constructor():
this(null, null)\n  actual constructor(message: String?): this(message, null)\n
  actual constructor(cause:
Throwable?): this(undefi
ned, cause)\n}\n\n/*\n * Copyright 2010-2019 JetBrains s.r.o. Use of this source
code is
governed by the Apache 2.0 license\n * that can be found in the license/LICENS
E.txt file.\n
*\n\n@file:Suppress("UNUSED_PARAMETER")\n\npackage kotlin.js\n\n@kotl
in.internal.InlineOnly\n\ninternal
inline fun jsDeleteProperty(obj: Any, property: Any) {\n  js("delete
obj[property]")\n}\n\n@kotl
in.internal.InlineOnly\n\ninternal inline fun jsBitwiseOr(lhs: Any?, rhs: Any?):
Int =\njs("lhs | rhs").unsafeCast<Int>()", /*\n * Copyright 2010-2018 JetBrains
s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license
that can be found in the
license/LICENSE.txt file.\n *\n\npackage kotlin.math\n\n/**\n * Returns this value
with the sign bit same as of the
[sign] value.\n *\n * If [sign] is `NaN` the sign of the result is undefined.\n
*\n\n@SinceKotlin("1.2")\n\npublic actual
fun Double.withSign(sign: Double): Double {\n  val thisSignBit =
js("Kotlin").doubleSignBit(this).unsafeCast<Int>()\n  val newSignBit =
js("Kotlin").doubleSignBit(sign).unsafeCast<Int>()\n  return if (thisSignBit ==
newSignBit) this else -
this\n}\n\n/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of
this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n
*\n\npackage kotlin\n\n/**\n * Returns a bit representation of the specified float
ing-point value as [Long]\n *
according to the IEEE 754 floating-point "double format" bit layout.\n
*\n\n@SinceKotlin("1.2")\n\n@library("doubleToBits")\n\npublic actual fun
Double.toBits(): Long =
definedExternally\n\n/**\n * Returns a bit representation of the specified float
ing-point value as [Long]\n *
according to the IEEE 754 floating-point "double format" bit layout,\n * preserv
ing `NaN` values exact layout.\n
*\n\n@SinceKotlin("1.2")\n\n@library("doubleToRawBits")\n\npublic actual fun
Double.toRawBits(): Long =
definedExternally\n\n/**\n * Returns the [Double] value corresponding to a given
bit representation.\n
*\n\n@SinceKotlin("1.2")\n\n@kotl
in.internal.InlineOnly\n\npublic actual inline fun Double.Companion.fromBits
(bits:
Long): Double = js("Kotlin").doubleFromBits(bits).unsafeCast<Double>()\n\n/**\n
 * Returns a bit representation

```

of the specified floating-point value as [Int] according to the IEEE 754 floating-point "single format" bit layout. Note that in Kotlin/JS [Float] range is wider than "single format" bit layout can represent, so some [Float] values may overflow, underflow or lose their accuracy after conversion to bits and back.

```

*\/n@SinceKotlin("1.2")\/n@library("floatToBits")\/npublic actual fun Float.toBits(): Int =
definedExternally\/n\/n**\/n * Returns a bit representation of the specified floating-point value as [Int] according
to the IEEE 754 floating-point "single format" bit layout, preserving `NaN` values exact layout. Note
that in Kotlin/JS [Float] range is wider than "single format" bit layout can represent, so some [Float] values
may overflow, underflow or lose their accuracy after conversion to bits and back.
*\/n@SinceKotlin("1.2")\/n@library("floatToRawBits")\/npublic actual fun Float.toRawBits(): Int =
definedExternally\/n\/n**\/n * Returns the [Float] value corresponding to a given bit representation.
*\/n@SinceKotlin("1.2")\/n@kotlin.internal.InlineOnly\/npublic actual inline fun Float.Companion.fromBits(bits:
Int): Float =
js("Kotlin").floatFromBits(bits).unsafeCast<Float>()\/n\/n@Suppress("NOTHING_TO_INLINE")\/ninternal
inline fun Long(low: Int, high: Int) = js("Kotlin").Long.fromBits(low, high).unsafeCast<Long>()\/ninternal inline
val Long.low: Int get() = this.asDynamic().getLowBits().unsafeCast<Int>()\/ninternal inline val Long.high: Int get()
= this.asDynamic().getHighBits().unsafeCast<Int>()\/n", /*\/n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin
Programming Language contributors. Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.
*\/nimport kotlin.reflect.KClass\/n@PublishedApi\/ninternal fun <T :
Annotation> KClass<*>.findAssociatedObject(@Suppress("UNUSED_PARAMETER") annotationClass:
KClass<T>): Any? {\/n // This API is not supported in js-v1. Return `null` to be source-compatible with js-ir.
return null\/n}\/n", /*\/n * Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming Language contributors.
Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.
*\/npackage kotlin.text\/n\/n**\/n * Returns a string representation of this [Long] value in the specified [radix].
*\/n * @throws IllegalArgumentException when [radix] is not a valid radix for number to string conversion.
*\/n@SinceKotlin("1.2")\/npublic actual fun Long.toString(radix: Int): String =
asDynamic().toString(checkRadix(radix)), /*\/n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming
Language contributors. Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.
*\/npackage kotlin.text\/n\/n\/n// NOTE: THIS FILE IS AUTO-GENERATED by the
GenerateUnicodeData.kt\/n// See: https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\/n\/n// 1343 ranges
totally\/nprivate object Category {\/n val decodedRangeStart: IntArray\/n val decodedRangeCategory: IntArray\/n
\/n init {\/n val toBase64 =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+\/"\/n val fromBase64 =
IntArray(128)\/n for (i in toBase64.indices) {\/n fromBase64[toBase64[i].code] = i\/n }\/n //
rangeStartDiff.length = 1482\/n val rangeStartDiff =
"gBCFEDCKCDCaDDaDBhBCEEDDDDDDEDXBHYBH5BRwBGDCHDCIDFHDFDCDEIRTEE7BGHDDJI
CBbSEMOFGERwDEDDDDDECEFCRBjBFDfCYFFCCzBvBjBBFC3BoHdBmBDGpBDDcTBjBEECLGDFC
LDCgBBKVKEDiDDHCFECECKCEODBeBc5CLBOKhBJDDDDWEBHFfCfCPBZDEL1BVBSLPBgBB2BDB
DICFBHKCKCPDBHEDWBHEDDDDEDEDIBDGDCCKCGDDDCGEGCCWBFMDDCEDDDCHDDHKDDBK
DBHFfCWBFgFDBDDFEDBPDDKCHBGDCHEDWBFgFDCEDEDBHDDGDCKCGJEGDBFDDFDDDDDME
FDBFDfCGBOKDFDFDCGfCXBQDDDDDBEGEDFDDKHBHDDGfCXBKBFCEfCFCHCECKDNCCHFfC
oBEDECFDDDDHDCKJBGDCSDYBJEHBfDDEBIGKDCMuBFHEBGBIBKcKBFfBfBXEIFJDFDGCKCEgB
BDPEDGKKGECIBkBEObDFFLBkBBIBEFFECIBrBCEBEGDBKGGDDDDDDCHDENDCFEKDDIBDDFrBCD
pKBECGEECPBBEChBBECGEECPB5BBECjCCDJUDQKG2CCGDsTCRbaCDrCDDIHNBEDLSDCJSCMLFC
CM0BDHGFLBFDDKKGKGEFDDBKgjBB1BHfChBDFmCKfDDDDDDCGDCFDKcFLsBEaGKBdiBXDDD1
BDGDEIGJEKKGHGBGCMf/BEBvBCEDDFHEKHkJJDDDeDDGDkSBFEDCIEkBIICDFKDDKeGCJHrBCDI
IDBNBHEBEfDBfSb/BNBiBIB6BBf1EiIdJIGCGCIIIIIGCGCIIIIIOCIIIIIDFEDDBFEDDDDEBDIFDDFEDBLF
GCEEICfBJCDEDCLDKBfBKCCGDDKDDNDgBQNEBDMPPFDEDEBFFHECEBEEDFBEDDQjBCEDEFFC
CJHBeEEfsIIEUCHCxCBeZoBGICZLV8BuCW3FBjB2BIvDB4HOesBFfCfKQgJjEW/BEgBCiWbVCCGnBCgBBp

```

```

DvBBuBEDBHEFGCCjDCGEDCFcfIBDDF4BHCObXJHBHBHBHBHBHBHBgBCECGHGEDI FBKCEDM
EtBaB5CM2GaMEDDCKCGFCJEDFDDDC2CDDDB6CDCFrBB+CDEKgbkBMQfBKeIBPgBKnBPgKguGgC9
vUDVB3jBD3BJoBGCsIBDQKCUuBDDKCCmCKCGIXJcNC/BBHGKDECEVFBEMCEEbqBDDGDFDXD
CEBDGEG0BEICyBQCICKGSGDEBKcICXLCLBdDDBvBDECCDNCKECFCJKFBpBFEDCJDBICCKCEQBG
DDByBEDCEFBYDCLDDCKGCGJHBHBrBBEJDEwCjBIDCKGk9KMxExBEggCgoGuLcQDmBHMFFC
KBNBFBIsDQRrLCQgCC2BoBMCCQGEGQDCQDDDDFDGDECEEFBnEEBFEDCKCDaDDaDBFCKBtBCf
DGCGCFEDDDDCHECKDC"\n    val diff = decodeVarLenBase64(rangeStartDiff, fromBase64, 1342)\n    val
start = IntArray(diff.size + 1)\n    for (i in diff.indices) {\n        start[i + 1] = start[i] + diff[i]\n    }\n
decodedRangeStart = start\n    // rangeCategory.length = 2033\n    val rangeCategory =
\"PsY44a41W54UYJZYB14W7XC15WZPsYa84b19Zw8b85Lr7C44brlerrYBZBCZCiBiBiBhCiiBhChiBhiCBhh
ChiCihBhChCChiBhChiCIBCFhjCiBiBihDhiBhCCihBiBBhCCFCEbEbEb7EbGhCk7BixRkiCi4BRbh4BhRhCBR
BCiiBBCiBChiZCBCiBcGHhChCiBRBxxEYC40Rx8c6RGUm4GRFRFYRQZ44acG4wRYFEFGJYIIIGFIYGwc
GmkEmcGFJfI8cYxwFGFGRFGFRJFGkkcYkxRm6aFGEgmmEmEGRYRFGxxYFRFRFRGQGI FmIFIGIooGF
GFGYJ4EFmoIRFlxRlxRFRFRxIRxIFIRxmFIGxxIoxRomFRIRxIFmGRJFaL86F4mRxmGoRFRFRFRFIRxGIGR
xmGxmGmxRxGRFIRRjmmFIIGYRmmIRFIRIRFRFIRFxxGFIGmmRoxImxRFRII GmxRJ4aRFGxmIoRFlxRlxR
FRFIRFxxGII moGmmRxoIxoIGRmmIRxIFmGRJ8FLRxmFFRFRIRIRxxFIRlxRxlFRFRFRooGRIOoRomRxFRIR
JLc8aRmoIoGFIRIRFRFRImGmoIoRGRGRxmGFRII GmxRJR YL8IGooYFIIRIRFRFRFRmIIxGooRGRIRlxFG
RJxIFRGIFIRIRFImGIGxIoRomF8xRxxFIILFGRJLcFxmIoRFRFRFxIRFRxxGxxIoGmmRRIRJxxIoYRFIIGG
RaFEGYJYRxIFRFRFIRFII GGlxRFxEGRJRFRFcY84c8mGcJL8G1WIFRFRGIGmmYFGRGRcGc88RYcYRFIGI
GmmIomGFJYFooGmIFII GmmFIFIFGFmoIGIomFJIm8cBhRRxxBC4ECFRFRFIRFRFRFRFRFRFRFRFRFRFR
FRGYLRfcRBRCxxUF8YFMF1WRFYKFRFRFGFRGYRFGFRFIRIRGRFmmIGIooGGY44E46FmxRJRLRY44
U44GmmQRJRFEFRFGFGRFRFxGmoIoGmoIoxRxxIoGIGRxxc4YJFRFRFRFRJLRcFmmIomRx4YFoGG
mRomIGIGmxRJRJRYEYRGmmHRGIFmIGmIooGFRJYcGcRmmIFomGmmIomGmIFJfmoGooGGIRYFII GIG
RYJRfJFEYCRBRBYRGYGIGFGFIIGomGFRCECECEGRGHCCiBCBCRBCBCBCRBCxBCBCRCD CDCD
CiiRBj7CbCiiRBj7b7iCiiRxiCBRbCBbxxCiiRBj7bRMQUY9+V9+VYtOQMY9eY43X44Z1WY54XYMQRQRER
LZ12ELZ12RERaRGHGHR88B88BihBhiChhC8hcZbC8BB8CBcFi8cihBZBC8Z8CLKhCkR8cRZcZc88ZcZc85
Z8ZcZc1WcZc1WcZcZcZcRcRLcLcZcZcZcZc1WlCZ1WZ1WZcZ1WZ1WZ1WZcZcZcRcRcBRcixBBCiBBihC
CEBhCCChCGhCRY44LCiRRxxCFRkYRGFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFR
48cRcRcRcRcRs4Y48EIK1Wc1W12U2cKGooUE88KqqEl4c8RFxxGm7bkkFUF4kEkFRFRFRx8cLcFcRfRLcLc
LcLcLcLcFRFEFRcRFEYFEYFJRhClmHnnYG4EhCEGFKGYRbEbCCiBECiBhCk7bhCIBihCiBBCBhCRhiBh
hCCRhiFkkCFIIGIIGIFooGmIcGRL88aRFYRIFIGRYJRGFY14FGJFGYFGIRYFRGIFmoIGIGIYxEJRYFmEFJ
FRFGmoImoIGRFGFmIRJRYFEFcloGIFmI GmIFGFI GFRII EYFomGo4YIkEoGRFRFRFRFRFRFRcBECK7bRCFo
oG4oGRJRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFRFR
44U1WY50Z5R46YRFRFRxxQY44a41W54UYJZYB14W7XC15WZ12YYFEFEFRFRFRFRFlxRIIRxxa65b86axcZc
RQcR"\n    decodedRangeCategory = decodeVarLenBase64(rangeCategory, fromBase64, 1343)\n
}\n}\n\nprivate fun categoryValueFrom(code: Int, ch: Int): Int {\n    return when {\n        code < 0x20 -> code\n        code < 0x400 -> if ((ch and 1) == 1) code shr 5 else code and 0x1f\n            else ->\n                when (ch % 3) {\n                    2 -> code shr 10\n                    1 -> (code shr 5) and 0x1f\n                    else -> code and 0x1f\n                }\n            }\n    }\n}\n\n/*\n * Returns the Unicode general category of this character as an Int.\n */\ninternal fun Char.getCategoryValue(): Int
{\n    val ch = this.code\n    val index = binarySearchRange(Category.decodedRangeStart, ch)\n    val start =
Category.decodedRangeStart[index]\n    val code = Category.decodedRangeCategory[index]\n    val value =
categoryValueFrom(code, ch - start)\n    return if (value == 17) CharCategory.UNASSIGNED.value else
value\n}\n\ninternal fun decodeVarLenBase64(base64: String, fromBase64: IntArray, resultLength: Int): IntArray
{\n    val result = IntArray(resultLength)\n    var index = 0\n    var int = 0\n    var shift = 0\n    for (char in base64)
{\n        val sixBit = fromBase64[char.code]\n        int = int or ((sixBit and 0x1f) shl shift)\n        if (sixBit < 0x20)
{\n            result[index++] = int\n            int = 0\n            shift = 0\n        } else {\n            shift += 5\n        }\n    }\n    return result\n}\n\n/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n *
Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n

```

```

*/\n\npackage kotlin.collections\n\n/\n// NOTE: THIS FILE IS AUTO-GENERATED by the
GenerateStandardLib.kt\n// See: https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\nimport
kotlin.js.*\nimport kotlin.ranges.contains\nimport kotlin.ranges.reversed\n\n/*\n * Reverses elements in the list in-
place.\n *\npublic actual fun <T> MutableList<T>.reverse(): Unit {\n    val midPoint = (size / 2) - 1\n    if
(midPoint < 0) return\n    var reverseIndex = lastIndex\n    for (index in 0..midPoint) {\n        val tmp = this[index]\n
        this[index] = this[reverseIndex]\n        this[reverseIndex] = tmp\n        reverseIndex--\n    }\n}\n\n"/\n *
Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is
governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n *\n\npackage
kotlin.text\n\n/\n// NOTE: THIS FILE IS AUTO-GENERATED by the GenerateUnicodeData.kt\n// See:
https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\n// 37 ranges totally\nprivate object Digit {\n
internal val rangeStart = intArrayOf(\n    0x0030, 0x0660, 0x06f0, 0x07c0, 0x0966, 0x09e6, 0x0a66, 0x0ae6,
0x0b66, 0x0be6, 0x0c66, 0x0ce6, 0x0d66, 0x0de6, 0x0e50, 0x0ed0, 0x0f20, 0x1040, 0x1090, 0x17e0, \n
0x1810, 0x1946, 0x19d0, 0x1a80, 0x1a90, 0x1b50, 0x1bb0, 0x1c40, 0x1c50, 0xa620, 0xa8d0, 0xa900, 0xa9d0,
0xa9f0, 0xaa50, 0xabf0, 0xff10, \n    )\n}\n\n/*\n * Returns the index of the largest element in [array] smaller or
equal to the specified [needle],\n * or -1 if [needle] is smaller than the smallest element in [array].\n *\ninternal fun
binarySearchRange(array: IntArray, needle: Int): Int {\n    var bottom = 0\n    var top = array.size - 1\n    var middle
= -1\n    var value = 0\n    while (bottom <= top) {\n        middle = (bottom + top) / 2\n        value = array[middle]\n
        if (needle > value)\n            bottom = middle + 1\n        else if (needle == value)\n            return middle\n
        else\n            top = middle - 1\n    }\n    return middle - (if (needle < value) 1 else 0)\n}\n\n/*\n * Returns an integer
from 0..9 indicating the digit this character represents,\n * or -1 if this character is not a digit.\n *\ninternal fun
Char.digitToIntImpl(): Int {\n    val ch = this.code\n    val index = binarySearchRange(Digit.rangeStart, ch)\n    val
diff = ch - Digit.rangeStart[index]\n    return if (diff < 10) diff else -1\n}\n\n/*\n * Returns `true` if this character
is a digit.\n *\ninternal fun Char.isDigitImpl(): Boolean {\n    return digitToIntImpl() >= 0\n}\n\n"/\n * Copyright
2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed
by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n *\n\npackage kotlin.text\n\n/\n//
NOTE: THIS FILE IS AUTO-GENERATED by the GenerateUnicodeData.kt\n// See:
https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\n// 222 ranges totally\nprivate object Letter {\n
val decodedRangeStart: IntArray\n    val decodedRangeLength: IntArray\n    val decodedRangeCategory: IntArray\n
\n    init {\n        val toBase64 =
\n        \"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/\n        val fromBase64 =
\n        IntArray(128)\n        for (i in toBase64.indices) {\n            fromBase64[toBase64[i].code] = i\n        }\n
\n        //
rangeStartDiff.length = 356\n        val rangeStartDiff =
\n        \"hCgBpCQGYHZH5BRpBPPPPPRMP5BPPICPP6BkEPPPPcXPzBvBrB3BOiDoBHwD+E3DauCnFmBmB2D
6E1BIBTiBmBIBP5BhBiBrBvBjBqBnBPRtBiCmCtBIB0BmB5BiB7BmBgEmChBzGCoEoGVpBsFrhBPqKQ2B
wBYoFgB4CJuTiEvBuCuDrF5DgEgFIJ1DgFmBQtBsBRGsB+BPiBID1EijDPRPPPPPPPPPGQSQS/DxENVNU+
B9zCwBwBPPCkDPNnBPqDYY1R8B7FkFgTgwGgwUwmBgKwBuBScmEP/BPPPPPRpB8B7F1B/ErBqC6B7B
iBmBfQsBUwCw/KwqIwLwETPcPjQgJxFgBIBsD\n        val diff = decodeVarLenBase64(rangeStartDiff,
fromBase64, 222)\n        val start = IntArray(diff.size)\n        for (i in diff.indices) {\n            if (i == 0) start[i] =
diff[i]\n            else start[i] = start[i - 1] + diff[i]\n        }\n        decodedRangeStart = start\n        //
rangeLength.length = 328\n        val rangeLength =
\n        \"aaMBXHYH5BRpBPPPPPRMP5BPPICPPzBDOOPPcXPzBvBjB3BOhDmBBpB7DoDYxB+EiBP1DoExBkB
QhBekBPmBgBhBctBiBMWOOXhCsBpBkBUV3Ba4BkB0DICgBXgBtD4FSdBfPhBPpKP0BvBXjEQ2CGsT8Dh
BtCqDpFvD1D3E0IrD2EkBJrBDOBsB+BPiBIB1EijDPPPPPPPPPPPGPPMNLsBNPNPKCvBvBPPCkDPBmBPh
DXXgD4B6FzEgDguG9vUtkB9JcuBSckEP/BPPPPPPBPf4FrBjEhBpC3B5BKaWPrBOWck/KsCuLqDHPbPxPsFt
EaaqDL\n        decodedRangeLength = decodeVarLenBase64(rangeLength, fromBase64, 222)\n        //
rangeCategory.length = 959\n        val rangeCategory =
\n        \"GFjgggUHGFFZZZmzpz5qB6s6020B60ptltB6smt2sB60mz22B1+vv+8BZZ5s2850BW5q1ymtB506smzBF3q1
q1qB1q1q1+Bgii4wDTm74g3KigxqM60q1q1Bq1o1q1BF1qlrqrBZ2q5wprBGFZWWZGHFsjiioLowgmOowjkw

```

```

CkgoiIk7ligGogiioBkwkiYkzj2oNoi+sbkwj04DghhkQ8wgiYkgoioDsgnkwC4gikQ//v+85BkwvoIsgoyI4yguI0whiw
Eowri4CoghsJowgqYowgm4DkwgsY/nwzPowhmYkg6wI8yggZswikwHgxgmIoxgqYkwgk4DKxgmIkgoioBsgsso
BgzyI8g9gL8g9kI0wgvJoxgkoC0wgioFkw/wI0w53iF4gioYowjmgBHgq1qkgwBF1q1q8qBHwghuIwghyKk0go
QkwgoQk3goQHGFHkyg0pBgxj6IoinkxDswno7Ikwhz9Bo0gioB8z48Rwli0xN0mpjoX8w78pDwltoqKHFEGGwwg
sIHFH3q1q16BFHWFZ1q10q1B2qlwq1B1q10q1B2q1yq1B6q1gq1Biq1qhxBir1qp1Bqt1q1qB1g1q1+B//3q16B///q
1qBH/qlq9Bholqq9B1i00a1q10qD1op1HkwmiGigiy6Cptogq1Bixo1kDq7/j00B2qgoBWGFm1lz50B6s5q1+BG
WhggzhwBFFhgk4//Bo2jigE8wguI8wguI8wgugUog1qoB4qjmIwwi2KkgYHHH4IBgiFWkgIWoghssMmz5smrBZ
3q1y50B5sm7gzBtz1smzB5smz50BqzqtmzB5sgzqzBF2/9//5BowgoIwmnkzPkwgk4C8ys65BkgoqI0wgy6FghquZo
2giY0ghiIsgH24B4ghsQ8QF/v1q1OFs0O8iCHHF1qggz/B8wg6Iznv+//B08QgohsjK0QGfK7hsQ4gB"

decodedRangeCategory = decodeVarLenBase64(rangeCategory, fromBase64, 222)
}

Returns
`true` if this character is a letter.
internal fun Char.isLetterImpl(): Boolean { return getLetterType() != 0 }

Returns `true` if this character is a lower case letter, or it has contributory property
Other_Lowercase.
internal fun Char.isLowerCaseImpl(): Boolean { return getLetterType() == 1 ||
code.isOtherLowercase() }

Returns `true` if this character is an upper case letter, or it has contributory
property Other_Uppercase.
internal fun Char.isUpperCaseImpl(): Boolean { return getLetterType() == 2 ||
code.isOtherUppercase() }

Returns
- `1` if the character is a lower case letter,
- `2` if the character is an upper case letter,
- `3` if the character is a letter but not a lower or upper case letter,
- `0` otherwise.
private fun Char.getLetterType(): Int {
    val ch = this.code
    val index =
        binarySearchRange(Letter.decodedRangeStart, ch)
    val rangeStart = Letter.decodedRangeStart[index]
    val rangeEnd = rangeStart + Letter.decodedRangeLength[index] - 1
    val code = Letter.decodedRangeCategory[index]
    if (ch > rangeEnd) { return 0 }
    val lastTwoBits = code and 0x3
    if (lastTwoBits == 0) { // gap pattern
        var shift = 2
        var threshold = rangeStart
        for (i in 0..1) {
            threshold += (code shr shift) and 0x7f
            if (threshold > ch) { return 3 }
        }
        shift += 7
        threshold += (code shr shift) and 0x7f
        if (threshold > ch) { return 0 }
    }
    shift += 7
    return 3
}
if (code <= 0x7) { return lastTwoBits }
val distance = (ch - rangeStart)
val shift = if (code <= 0x1F) distance % 2 else distance
return (code shr (2 * shift)) and 0x3 }

"
/*
Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.
Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.

package kotlin.text

NOTE: THIS FILE IS AUTO-GENERATED by the GenerateUnicodeData.kt
See: https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib

private object OtherLowercase {
    internal val otherLowerStart = arrayOf(
        0x00aa, 0x00ba, 0x02b0, 0x02c0, 0x02e0,
        0x0345, 0x037a, 0x1d2c, 0x1d78, 0x1d9b, 0x2071, 0x207f, 0x2090, 0x2170, 0x24d0, 0x2c7c, 0xa69c, 0xa770,
        0xa7f8, 0xab5c,
    )
    internal val otherLowerLength = arrayOf(
        1, 1, 9, 2, 5, 1, 1, 63, 1, 37, 1, 1, 13,
        16, 26, 2, 2, 1, 2, 4,
    )
}

internal fun Int.isOtherLowercase(): Boolean {
    val index =
        binarySearchRange(OtherLowercase.otherLowerStart, this)
    return index >= 0 && this <
        OtherLowercase.otherLowerStart[index] + OtherLowercase.otherLowerLength[index]
}

"
/*
Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.
Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.

package kotlin.text

NOTE: THIS FILE IS AUTO-GENERATED by the GenerateUnicodeData.kt
See:
https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib

internal fun Int.isOtherUppercase(): Boolean {
    return this in 0x2160..0x216f || this in 0x24b6..0x24cf
}

"
/*
Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.
Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.

package kotlin.text

NOTE: THIS FILE IS AUTO-GENERATED by the GenerateStandardLib.kt
See:
https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib

import kotlin.js.*

Returns a character at the given [index] or throws an [IndexOutOfBoundsException] if the [index] is out of bounds of this char sequence.

@sample samples.collections.Collections.Elements.elementAt

public actual fun CharSequence.elementAt(index: Int): Char {
    return elementAtOrElse(index) { throw

```



```

return this@asList.lastIndexOf(element)\n    }\n }\n}\n\n/**\n * Returns a [List] that wraps the original
array.\n */\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic actual fun UByteArray.asList():
List<UByte> {\n    return object : AbstractList<UByte>(), RandomAccess {\n        override val size: Int get() =
this@asList.size\n        override fun isEmpty(): Boolean = this@asList.isEmpty()\n        override fun
contains(element: UByte): Boolean = this@asList.contains(element)\n        override fun get(index: Int): UByte {\n
            AbstractList.checkElementIndex(index, size)\n            return this@asList[index]\n        }\n        override fun
indexOf(element: UByte): Int {\n            @Suppress("USELESS_CAST")\n            if ((element as Any?) !is
UByte) return -1\n            return this@asList.indexOf(element)\n        }\n        override fun lastIndexOf(element:
UByte): Int {\n            @Suppress("USELESS_CAST")\n            if ((element as Any?) !is UByte) return -1\n
return this@asList.lastIndexOf(element)\n        }\n }\n }\n}\n\n/**\n * Returns a [List] that wraps the original array.\n
*/\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic actual fun UShortArray.asList(): List<UShort>
{\n    return object : AbstractList<UShort>(), RandomAccess {\n        override val size: Int get() = this@asList.size\n
        override fun isEmpty(): Boolean = this@asList.isEmpty()\n        override fun contains(element: UShort):
Boolean = this@asList.contains(element)\n        override fun get(index: Int): UShort {\n
            AbstractList.checkElementIndex(index, size)\n            return this@asList[index]\n        }\n        override fun
indexOf(element: UShort): Int {\n            @Suppress("USELESS_CAST")\n            if ((element as Any?) !is
UShort) return -1\n            return this@asList.indexOf(element)\n        }\n        override fun lastIndexOf(element:
UShort): Int {\n            @Suppress("USELESS_CAST")\n            if ((element as Any?) !is UShort) return -1\n
return this@asList.lastIndexOf(element)\n        }\n }\n }\n}\n\n"/*\n * Copyright 2010-2021 JetBrains s.r.o. and
Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.text\n\n// NOTE: THIS FILE IS AUTO-
GENERATED by the GenerateUnicodeData.kt\n// See:
https://github.com/JetBrains/kotlin/tree/master/libraries/stdlib\n\n// 9 ranges totally\n\n/**\n * Returns `true` if this
character is a whitespace.\n */\n@internal fun Char.isWhitespaceImpl(): Boolean {\n    val ch = this.code\n    return ch
in 0x0009..0x000d\n        || ch in 0x001c..0x0020\n        || ch == 0x00a0\n        || ch > 0x1000 && (\n
ch == 0x1680\n        || ch in 0x2000..0x200a\n        || ch == 0x2028\n        || ch == 0x2029\n
|| ch == 0x202f\n        || ch == 0x205f\n        || ch == 0x3000\n        )\n}\n\n"/*\n * Copyright 2010-2020
JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin\n\npublic actual fun
interface Comparator<T> {\n    @JsName("compare")\n    public actual fun compare(a: T, b: T): Int\n}\n\n"/*\n *
Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is
governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage
kotlin.js\n\nimport kotlin.annotation.AnnotationTarget.*\n\n@Target(FUNCTION)\n@Deprecated("Use inline
extension function with body using dynamic")\npublic annotation class
nativeGetter\n\n@Target(FUNCTION)\n@Deprecated("Use inline extension function with body using
dynamic")\npublic annotation class nativeSetter\n\n@Target(FUNCTION)\n@Deprecated("Use inline extension
function with body using dynamic")\npublic annotation class nativeInvoke\n\n@Target(CLASS, FUNCTION,
PROPERTY)\n@internal annotation class library(public val name: String = "")\n\n@Target(CLASS)\n@internal
annotation class marker\n\n/**\n * Gives a declaration (a function, a property or a class) specific name in
JavaScript.\n */\n * This may be useful in the following cases:\n * * There are two functions for which the
compiler gives same name in JavaScript, you can\n * mark one with `@JsName(...)` to prevent the compiler from
reporting error.\n * * You are writing a JavaScript library in Kotlin. The compiler produces mangled names\n *
for functions with parameters, which is unnatural for usual JavaScript developer.\n * You can put `@JsName(...)`
on functions you want to be available from JavaScript.\n * * For some reason you want to rename declaration, e.g.
there's common term in JavaScript\n * for a concept provided by the declaration, which is uncommon in Kotlin.\n
*/\n * Example:\n * ```\n kotlin\n * class Person(val name: String) {\n *     fun hello() {\n *         println("Hello
$name!")\n *     }\n *     @JsName("helloWithGreeting")\n *     fun hello(greeting: String) {\n *
println(" $greeting $name!")\n *     }\n * }\n * ```\n */\n * @property name the name which compiler uses both for

```

declaration itself and for all references to the declaration.

It's required to denote a valid JavaScript identifier.

`@Retention(AnnotationRetention.BINARY)`  
`@Target(CLASS, FUNCTION, PROPERTY, CONSTRUCTOR, PROPERTY_GETTER, PROPERTY_SETTER)`  
 public actual annotation class JsName(actual val name: String)

Denotes an `external` declaration that must be imported from native JavaScript library.

The compiler produces the code relevant for the target module system, for example, in case of CommonJS, it will import the declaration via the `require(...)` function.

The annotation can be used on top-level external declarations (classes, properties, functions) and files.

In case of file (which can't be `external`) the following rule applies: all the declarations in the file must be `external`. By applying `@JsModule(...)` on a file you tell the compiler to import a JavaScript object that contain all the declarations from the file.

Example:

```
kotlin
@JsModule("jquery")
external abstract class JQuery() {
    // some declarations here
}
@JsModule("jquery")
external fun JQuery(element: Element): JQuery
@property import name of a module to import declaration from.
```

It is not interpreted by the Kotlin compiler, it's passed as is directly to the target module system.

`@see JsNonModule`

`@Retention(AnnotationRetention.BINARY)`  
`@Target(CLASS, PROPERTY, FUNCTION, FILE)`  
 public annotation class JsModule(val import: String)

Denotes an `external` declaration that can be used without module system.

By default, an `external` declaration is available regardless your target module system.

However, by applying [JsModule] annotation you can make a declaration unavailable to `plain` module system.

Some JavaScript libraries are distributed both as a standalone downloadable piece of JavaScript and as a module available as an npm package.

To tell the Kotlin compiler to accept both cases, you can augment [JsModule] with the `@JsNonModule` annotation.

For example:

```
kotlin
@JsModule("jquery")
@JsNonModule
@JsName("$")
external abstract class JQuery() {
    // some declarations here
}
@JsModule("jquery")
@JsNonModule
@JsName("$")
external fun JQuery(element: Element): JQuery
@see JsModule
```

`@Retention(AnnotationRetention.BINARY)`  
`@Target(CLASS, PROPERTY, FUNCTION, FILE)`  
 public annotation class JsNonModule

Adds prefix to `external` declarations in a source file.

JavaScript does not have concept of packages (namespaces). They are usually emulated by nested objects.

The compiler turns references to `external` declarations either to plain unprefix names (in case of `plain` modules) or to plain imports.

However, if a JavaScript library provides its declarations in packages, you won't be satisfied with this.

You can tell the compiler to generate additional prefix before references to `external` declarations using the `@JsQualifier(...)` annotation.

Note that a file marked with the `@JsQualifier(...)` annotation can't contain non-`external` declarations.

Example:

```
@file:JsQualifier("my.jsPackageName")
package some.kotlinPackage
external fun foo(x: Int)
external fun bar(): String
@property value the qualifier to add to the declarations in the generated code.
```

It must be a sequence of valid JavaScript identifiers separated by the `.` character.

Examples of valid qualifiers are: `foo`, `bar.Baz`, `\_.\$.f`.

`@see JsModule`

`@Retention(AnnotationRetention.BINARY)`  
`@Target(AnnotationTarget.FILE)`  
 public annotation class JsQualifier(val value: String)

Exports top-level declaration on JS platform.

Compiled module exposes declarations that are marked with this annotation without name mangling.

This annotation can be applied to either files or top-level declarations.

It is currently prohibited to export the following kinds of declarations:

- expect declarations
- inline functions with reified type parameters
- suspend functions
- secondary constructors without `@JsName`
- extension properties
- enum classes
- annotation classes

Signatures of exported declarations must only contain `exportable` types:

- dynamic
- Any
- String
- Boolean
- Byte
- Short
- Int
- Float
- Double
- BooleanArray
- ByteArray
- ShortArray
- IntArray
- FloatArray
- DoubleArray
- Array<exportable-type>
- Function types with exportable parameters and return types
- external or @JsExport classes and interfaces
- Nullable counterparts of types above
- Unit return type. Must not be nullable

This annotation is experimental, meaning that restrictions mentioned above are subject to change.

`@ExperimentalJsExport`  
`@Retention(AnnotationRetention.BINARY)`  
`@Target(CLASS, PROPERTY, FUNCTION, FILE)`  
`@SinceKotlin("1.3")`  
 public actual annotation class JsExport

Copyright 2010-

2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n \* Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n \*/\n\npackage kotlin.jvm\n\n// these are used in common generated code in stdlib\n\n// TODO: find how to deprecate these ones\n\n@Target(AnnotationTarget.FIELD)\n@Retention(AnnotationRetention.SOURCE)\npublic actual annotation class Volatile\n\n@Target(AnnotationTarget.FUNCTION, AnnotationTarget.PROPERTY\_GETTER, AnnotationTarget.PROPERTY\_SETTER)\n@Retention(AnnotationRetention.SOURCE)\npublic actual annotation class Synchronized\n\n"/\*\n \* Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n \* Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n \*/\n\npackage kotlin.collections\n\n/\*\*\n \* Provides a skeletal implementation of the [MutableCollection] interface.\n \*/\n \* @param E the type of elements contained in the collection. The collection is invariant in its element type.\n \*/\npublic actual abstract class AbstractMutableCollection<E> protected actual constructor() : AbstractCollection<E>(), MutableCollection<E> {\n\n actual abstract override fun add(element: E): Boolean\n\n actual override fun remove(element: E): Boolean {\n checkIsMutable()\n val iterator = iterator()\n while (iterator.hasNext()) {\n if (iterator.next() == element) {\n iterator.remove()\n return true\n }\n }\n return false\n }\n\n actual override fun addAll(elements: Collection<E>): Boolean {\n checkIsMutable()\n var modified = false\n for (element in elements) {\n if (add(element)) modified = true\n }\n return modified\n }\n\n actual override fun removeAll(elements: Collection<E>): Boolean {\n checkIsMutable()\n return (this as MutableIterable<E>).removeAll { it in elements }\n }\n\n actual override fun retainAll(elements: Collection<E>): Boolean {\n checkIsMutable()\n return (this as MutableIterable<E>).removeAll { it !in elements }\n }\n\n actual override fun clear(): Unit {\n checkIsMutable()\n val iterator = this.iterator()\n while (iterator.hasNext()) {\n iterator.next()\n iterator.remove()\n }\n }\n\n @Deprecated("Provided so that subclasses inherit this function", level = DeprecationLevel.HIDDEN)\n @JsName("toJSON")\n protected fun toJSON(): Any = this.toArray()\n\n /\*\*\n \* This method is called every time when a mutating method is called on this mutable collection.\n \* Mutable collections that are built (frozen) must throw `UnsupportedOperationException`.\n \*/\n internal open fun checkIsMutable(): Unit {\n }\n\n"/\*\n \* Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n \* Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n \*/\n\n/\*\*\n \* Based on GWT AbstractList\n \*/\n \* Copyright 2007 Google Inc.\n \*/\n\npackage kotlin.collections\n\n/\*\*\n \* Provides a skeletal implementation of the [MutableList] interface.\n \*/\n \* @param E the type of elements contained in the list. The list is invariant in its element type.\n \*/\npublic actual abstract class AbstractMutableList<E> protected actual constructor() : AbstractMutableCollection<E>(), MutableList<E> {\n\n protected var modCount: Int = 0\n\n abstract override fun add(index: Int, element: E): Unit\n\n abstract override fun removeAt(index: Int): E\n\n abstract override fun set(index: Int, element: E): E\n\n /\*\*\n \* Adds the specified element to the end of this list.\n \* \* @return `true` because the list is always modified as the result of this operation.\n \*/\n actual override fun add(element: E): Boolean {\n checkIsMutable()\n add(size, element)\n return true\n }\n\n actual override fun addAll(index: Int, elements: Collection<E>): Boolean {\n AbstractList.checkPositionIndex(index, size)\n checkIsMutable()\n var \_index = index\n var changed = false\n for (e in elements) {\n add(\_index++, e)\n changed = true\n }\n return changed\n }\n\n actual override fun clear() {\n checkIsMutable()\n removeRange(0, size)\n }\n\n actual override fun removeAll(elements: Collection<E>): Boolean {\n checkIsMutable()\n return removeAll { it !in elements }\n }\n\n actual override fun retainAll(elements: Collection<E>): Boolean {\n checkIsMutable()\n return removeAll { it !in elements }\n }\n\n\n actual override fun iterator(): MutableIterator<E> = IteratorImpl()\n\n actual override fun contains(element: E): Boolean = indexOf(element) >= 0\n\n actual override fun indexOf(element: E): Int {\n for (index in 0..lastIndex) {\n if (get(index) == element) {\n return index\n }\n }\n return -1\n }\n\n actual override fun lastIndexOf(element: E): Int {\n for (index in lastIndex downTo 0) {\n if (get(index) == element) {\n return index\n }\n }\n return -1\n }\n\n actual override fun listIterator():

```

MutableListIterator<E> = listIterator(0)\n  actual override fun listIterator(index: Int): MutableListIterator<E> =
ListIteratorImpl(index)\n\n  actual override fun subList(fromIndex: Int, toIndex: Int): MutableList<E> =
SubList(this, fromIndex, toIndex)\n  /**\n   * Removes the range of elements from this list starting from
[fromIndex] and ending with but not including [toIndex].\n   */\n  protected open fun removeRange(fromIndex:
Int, toIndex: Int) {\n    val iterator = listIterator(fromIndex)\n    repeat(toIndex - fromIndex) {\n
iterator.next()\n    iterator.remove()\n  }\n}\n  /**\n   * Compares this list with another list instance
with the ordered structural equality.\n   */\n   * @return true, if [other] instance is a [List] of the same size, which
contains the same elements in the same order.\n   */\n  override fun equals(other: Any?): Boolean {\n    if (other
=== this) return true\n    if (other !is List<*>) return false\n\n    return AbstractList.orderedEquals(this, other)\n
  }\n  /**\n   * Returns the hash code value for this list.\n   */\n  override fun hashCode(): Int =
AbstractList.orderedHashCode(this)\n\n  private open inner class IteratorImpl : MutableListIterator<E> {\n    /**
the index of the item that will be returned on the next call to [next]()\n    */\n    protected var index = 0\n    /** the
index of the item that was returned on the previous call to [next]()\n    */\n    * or [ListIterator.previous]() (for
`ListIterator`),\n    */\n    * -1 if no such item exists\n    */\n    protected var last = -1\n\n    override fun
hasNext(): Boolean = index < size\n\n    override fun next(): E {\n      if (!hasNext()) throw
NoSuchElementException()\n      last = index++\n      return get(last)\n    }\n\n    override fun remove()
{\n      check(last != -1) { \"Call next() or previous() before removing element from the iterator.\" }\n\n
removeAt(last)\n      index = last\n      last = -1\n    }\n  }\n  /**\n   * Implementation of
`MutableListIterator` for abstract lists.\n   */\n  private inner class ListIteratorImpl(index: Int) : IteratorImpl(),
MutableListIterator<E> {\n    init {\n      AbstractList.checkPositionIndex(index,
this@AbstractMutableList.size)\n      this.index = index\n    }\n\n    override fun hasPrevious(): Boolean =
index > 0\n\n    override fun nextIndex(): Int = index\n\n    override fun previous(): E {\n      if
(!hasPrevious()) throw NoSuchElementException()\n      last = --index\n      return get(last)\n    }\n\n
override fun previousIndex(): Int = index - 1\n\n    override fun add(element: E) {\n      add(index, element)\n
      index++\n      last = -1\n    }\n\n    override fun set(element: E) {\n      check(last != -1) { \"Call
next() or previous() before updating element value with the iterator.\" }\n      set(last, element)\n    }\n
  }\n\n  private class SubList<E>(private val list: AbstractMutableList<E>, private val fromIndex: Int, toIndex: Int) :
AbstractMutableList<E>(), RandomAccess {\n    private var _size: Int = 0\n    init {\n
AbstractList.checkRangeIndexes(fromIndex, toIndex, list.size)\n      this._size = toIndex - fromIndex\n    }\n\n
    override fun add(index: Int, element: E) {\n      AbstractList.checkPositionIndex(index, _size)\n\n
list.add(fromIndex + index, element)\n      _size++\n    }\n\n    override fun get(index: Int): E {\n
AbstractList.checkElementIndex(index, _size)\n      return list[fromIndex + index]\n    }\n\n    override
fun removeAt(index: Int): E {\n      AbstractList.checkElementIndex(index, _size)\n      val result =
list.removeAt(fromIndex + index)\n      _size--\n      return result\n    }\n\n    override fun set(index: Int,
element: E): E {\n      AbstractList.checkElementIndex(index, _size)\n      return list.set(fromIndex + index,
element)\n    }\n\n    override val size: Int get() = _size\n\n    internal override fun checkIsMutable(): Unit =
list.checkIsMutable()\n  }\n}\n\n\"/*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\n/*\n * Based on GWT AbstractMap\n * Copyright 2007 Google Inc.\n */\n\npackage kotlin.collections\n\n/**\n * Provides a skeletal implementation of the [MutableMap] interface.\n */\n * The implementor is required to implement [entries] property, which should return mutable set of map entries, and
[put] function.\n */\n * @param K the type of map keys. The map is invariant in its key type.\n * @param V the type
of map values. The map is invariant in its value type.\n */\npublic actual abstract class AbstractMutableMap<K, V>
protected actual constructor() : AbstractMap<K, V>(), MutableMap<K, V> {\n  /**\n   * A mutable
[Map.Entry] shared by several [Map] implementations.\n   */\n  internal open class SimpleEntry<K, V>(override
val key: K, value: V) : MutableMap.MutableEntry<K, V> {\n    constructor(entry: Map.Entry<K, V>) :
this(entry.key, entry.value)\n\n    private var _value = value\n\n    override val value: V get() = _value\n\n
override fun setValue(newValue: V): V {\n      // Should check if the map containing this entry is mutable.\n

```

```

// However, to not increase entry memory footprint it might be worthwhile not to check it here and
// force subclasses that implement `build()` (freezing) operation to implement their own `MutableEntry`.
this@AbstractMutableMap.checkIsMutable()
    val oldValue = this._value
    this._value = newValue
    return oldValue
}
override fun hashCode(): Int = entryHashCode(this)
override fun toString(): String = entryToString(this)
override fun equals(other: Any?): Boolean = entryEquals(this, other)
}
// intermediate abstract class to workaround KT-43321
internal abstract class AbstractEntrySet<E : Map.Entry<K, V>, K, V> : AbstractMutableSet<E>() {
    final override fun contains(element: E): Boolean = containsEntry(element)
    abstract fun containsEntry(element: Map.Entry<K, V>): Boolean
    final override fun remove(element: E): Boolean = removeEntry(element)
    abstract fun removeEntry(element: Map.Entry<K, V>): Boolean
}
actual override fun clear() {
    entries.clear()
}
private var _keys: MutableSet<K>? = null
actual override val keys: MutableSet<K>
    get() {
        if (_keys == null) {
            _keys = object : AbstractMutableSet<K>() {
                override fun add(element: K): Boolean = throw UnsupportedOperationException("Add is not supported on keys")
                override fun clear() {
                    this@AbstractMutableMap.clear()
                }
                override operator fun contains(element: K): Boolean = containsKey(element)
                override operator fun iterator(): MutableIterator<K> {
                    val entryIterator = entries.iterator()
                    return object : MutableIterator<K> {
                        override fun hasNext(): Boolean = entryIterator.hasNext()
                        override fun next(): K = entryIterator.next().key
                        override fun remove() = entryIterator.remove()
                    }
                }
                override fun remove(element: K): Boolean {
                    checkIsMutable()
                    if (containsKey(element)) {
                        this@AbstractMutableMap.remove(element)
                        return true
                    }
                    return false
                }
                override val size: Int
                get() = this@AbstractMutableMap.size
            }
        }
        return _keys!!
    }
actual abstract override fun put(key: K, value: V): V?
actual override fun putAll(from: Map<out K, V>) {
    checkIsMutable()
    for ((key, value) in from) {
        put(key, value)
    }
}
private var _values: MutableCollection<V>? = null
actual override val values: MutableCollection<V>
    get() {
        if (_values == null) {
            _values = object : AbstractMutableCollection<V>() {
                override fun add(element: V): Boolean = throw UnsupportedOperationException("Add is not supported on values")
                override fun clear() = this@AbstractMutableMap.clear()
                override operator fun contains(element: V): Boolean = containsValue(element)
                override operator fun iterator(): MutableIterator<V> {
                    val entryIterator = entries.iterator()
                    return object : MutableIterator<V> {
                        override fun hasNext(): Boolean = entryIterator.hasNext()
                        override fun next(): V = entryIterator.next().value
                        override fun remove() = entryIterator.remove()
                    }
                }
                override val size: Int
                get() = this@AbstractMutableMap.size
            }
        }
        return _values!!
    }
actual override fun remove(key: K): V? {
    checkIsMutable()
    val iter = entries.iterator()
    while (iter.hasNext()) {
        val entry = iter.next()
        val k = entry.key
        if (key == k) {
            val value = entry.value
            iter.remove()
            return value
        }
    }
    return null
}
}
/**
 * This method is called every time when a mutating method is called on this mutable map.
 * Mutable maps that are built (frozen) must throw `UnsupportedOperationException`.
 */
internal open fun checkIsMutable(): Unit {
    // "/*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\npackage kotlin.collections\n\n/*\n * Provides a skeletal implementation of the [MutableSet] interface.\n *\n * @param E the type of elements contained in the set. The set is invariant in its element type.\n */\npublic actual abstract class AbstractMutableSet<E> protected actual constructor() : AbstractMutableCollection<E>(), MutableSet<E> {
    /**\n     * Compares this set with another set instance with the unordered structural equality.\n     *\n     * @return `true`, if [other] instance is a [Set] of the same size, all

```

```

elements of which are contained in this set.\n    */\n    override fun equals(other: Any?): Boolean {\n        if (other
=== this) return true\n        if (other !is Set<*>) return false\n        return AbstractSet.setEquals(this, other)\n    }\n\n    /**\n     * Returns the hash code value for this set.\n     */\n    override fun hashCode(): Int =
AbstractSet.unorderedHashCode(this)\n\n    }"/**\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\npackage kotlin.collections\n\n/**\n * Provides a [MutableList] implementation,
which uses a resizable array as its backing storage.\n */\n * This implementation doesn't provide a way to manage
capacity, as backing JS array is resizeable itself.\n * There is no speed advantage to pre-allocating array sizes in
JavaScript, so this implementation does not include any of the\n * capacity and "growth increment" concepts.\n */\n\npublic actual open class ArrayList<E> internal constructor(private var array: Array<Any?>) :
AbstractMutableList<E>(), MutableList<E>, RandomAccess {\n    private var isReadOnly: Boolean = false\n\n    /**\n     * Creates an empty [ArrayList].\n     */\n    public actual constructor() : this(emptyArray()) {\n\n    /**\n     * Creates an empty [ArrayList].\n     * @param initialCapacity initial capacity (ignored)\n     */\n    public actual
constructor(initialCapacity: Int) : this(emptyArray()) {\n\n    /**\n     * Creates an [ArrayList] filled from the
[elements] collection.\n     */\n    public actual constructor(elements: Collection<E>) :
this(elements.toArray<Any?>()) {\n\n    @PublishedApi\n    internal fun build(): List<E> {\n
checkIsMutable()\n        isReadOnly = true\n        return this\n    }\n\n    /** Does nothing in this ArrayList
implementation. */\n    public actual fun trimToSize() {\n\n    /** Does nothing in this ArrayList implementation.
*/\n    public actual fun ensureCapacity(minCapacity: Int) {\n\n    actual override val size: Int get() = array.size\n
@Suppress("UNCHECKED_CAST")\n    actual override fun get(index: Int): E = array[rangeCheck(index)] as E\n
actual override fun set(index: Int, element: E): E {\n        checkIsMutable()\n        rangeCheck(index)\n
@Suppress("UNCHECKED_CAST")\n        return array[index].apply { array[index] = element } as E\n    }\n\n
actual override fun add(element: E): Boolean {\n        checkIsMutable()\n        array.asDynamic().push(element)\n
modCount++\n        return true\n    }\n\n    actual override fun add(index: Int, element: E): Unit {\n
checkIsMutable()\n        array.asDynamic().splice(insertionRangeCheck(index), 0, element)\n        modCount++\n
}\n\n    actual override fun addAll(elements: Collection<E>): Boolean {\n        checkIsMutable()\n        if
(elements.isEmpty()) return false\n        array += elements.toArray<Any?>()\n        modCount++\n
return true\n    }\n\n    actual override fun addAll(index: Int, elements: Collection<E>): Boolean {\n
checkIsMutable()\n        insertionRangeCheck(index)\n        if (index == size) return addAll(elements)\n        if
(elements.isEmpty()) return false\n        when (index) {\n            size -> return addAll(elements)\n            0 -> array
= elements.toArray<Any?>() + array\n            else -> array = array.copyOfRange(0,
index).asDynamic().concat(elements.toArray<Any?>(), array.copyOfRange(index, size))\n        }\n\n
modCount++\n        return true\n    }\n\n    actual override fun removeAt(index: Int): E {\n        checkIsMutable()\n
rangeCheck(index)\n        modCount++\n        return if (index == lastIndex)\n            array.asDynamic().pop()\n
else\n            array.asDynamic().splice(index, 1)[0]\n    }\n\n    actual override fun remove(element: E): Boolean {\n
checkIsMutable()\n        for (index in array.indices) {\n            if (array[index] == element) {\n
array.asDynamic().splice(index, 1)\n                modCount++\n                return true\n            }\n        }\n
return false\n    }\n\n    override fun removeRange(fromIndex: Int, toIndex: Int) {\n        checkIsMutable()\n
modCount++\n        array.asDynamic().splice(fromIndex, toIndex - fromIndex)\n    }\n\n    actual override fun
clear() {\n        checkIsMutable()\n        array = emptyArray()\n        modCount++\n    }\n\n\n    actual override fun
indexOf(element: E): Int = array.indexOf(element)\n\n    actual override fun lastIndexOf(element: E): Int =
array.lastIndexOf(element)\n\n    override fun toString() = arrayToString(array)\n\n    @Suppress("UNCHECKED_CAST")\n    override fun <T> toArray(array: Array<T>): Array<T> {\n        if
(array.size < size) {\n            return toArray() as Array<T>\n        }\n        (this.array as
Array<T>).copyInto(array)\n        if (array.size > size) {\n            array[size] = null as T // null-terminate\n
}\n        return array\n    }\n\n    override fun toArray(): Array<Any?> {\n        return js("[]").slice.call(array)\n
}\n\n\n    internal override fun checkIsMutable() {\n        if (isReadOnly) throw UnsupportedOperationException()\n
}\n\n    private fun rangeCheck(index: Int) = index.apply {\n        AbstractList.checkElementIndex(index, size)\n
}

```

```

}\n\n private fun insertionRangeCheck(index: Int) = index.apply {\n    AbstractList.checkPositionIndex(index,
size)\n }},"/*\n * Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming Language contributors.\n *
Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*/\n\npackage kotlin.collections\n\ninternal fun <T> sortArrayWith(array: Array<out T>, comparison: (T, T) -> Int)
{\n    if (getStableSortingIsSupported()) {\n        array.asDynamic().sort(comparison)\n    } else {\n
mergeSort(array.unsafeCast<Array<T>>(), 0, array.lastIndex, Comparator(comparison))\n    }\n}\n\ninternal fun
<T> sortArrayWith(array: Array<out T>, comparator: Comparator<in T>) {\n    if (getStableSortingIsSupported())
{\n        val comparison = { a: T, b: T -> comparator.compare(a, b) }\n        array.asDynamic().sort(comparison)\n
    } else {\n        mergeSort(array.unsafeCast<Array<T>>(), 0, array.lastIndex, comparator)\n    }\n}\n\ninternal fun
<T> sortArrayWith(array: Array<out T>, fromIndex: Int, toIndex: Int, comparator: Comparator<in T>) {\n    if
(fromIndex < toIndex - 1) {\n        mergeSort(array.unsafeCast<Array<T>>(), fromIndex, toIndex - 1, comparator)\n
    }\n}\n\ninternal fun <T : Comparable<T>> sortArray(array: Array<out T>) {\n    if
(getStableSortingIsSupported()) {\n        val comparison = { a: T, b: T -> a.compareTo(b) }\n
array.asDynamic().sort(comparison)\n    } else {\n        mergeSort(array.unsafeCast<Array<T>>(), 0,
array.lastIndex, naturalOrder())\n    }\n}\n\nprivate var _stableSortingIsSupported: Boolean? = null\nprivate fun
getStableSortingIsSupported(): Boolean {\n    _stableSortingIsSupported?.let { return it }\n
_stableSortingIsSupported = false\n\n    val array = js("[]").unsafeCast<Array<Int>>()\n    // known
implementations may use stable sort for arrays of up to 512 elements\n    // so we create slightly more elements to
test stability\n    for (index in 0 until 600) array.asDynamic().push(index)\n    val comparison = { a: Int, b: Int -> (a
and 3) - (b and 3) }\n    array.asDynamic().sort(comparison)\n    for (index in 1 until array.size) {\n        val a =
array[index - 1]\n        val b = array[index]\n        if ((a and 3) == (b and 3) && a >= b) return false\n    }\n
_stableSortingIsSupported = true\n    return true\n}\n\nprivate fun <T> mergeSort(array: Array<T>, start: Int,
endInclusive: Int, comparator: Comparator<in T>) {\n    val buffer =
arrayOfNulls<Any?>(array.size).unsafeCast<Array<T>>()\n    val result = mergeSort(array, buffer, start,
endInclusive, comparator)\n    if (result !== array) {\n        for (i in start..endInclusive) array[i] = result[i]\n
    }\n}\n\n// Both start and end are inclusive indices.\nprivate fun <T> mergeSort(array: Array<T>, buffer: Array<T>,
start: Int, end: Int, comparator: Comparator<in T>): Array<T> {\n    if (start == end) {\n        return array\n    }\n
val median = (start + end) / 2\n    val left = mergeSort(array, buffer, start, median, comparator)\n    val right =
mergeSort(array, buffer, median + 1, end, comparator)\n    val target = if (left === buffer) array else buffer\n\n    //
Merge.\n    var leftIndex = start\n    var rightIndex = median + 1\n    for (i in start..end) {\n        when {\n
leftIndex <= median && rightIndex <= end -> {\n            val leftValue = left[leftIndex]\n            val rightValue
= right[rightIndex]\n            if (comparator.compare(leftValue, rightValue) <= 0) {\n                target[i] =
leftValue\n                leftIndex++\n            } else {\n                target[i] = rightValue\n
rightIndex++\n            }\n        }\n        leftIndex <= median -> {\n            target[i] = left[leftIndex]\n
leftIndex++\n        }\n        else /* rightIndex <= end */ -> {\n            target[i] = right[rightIndex]\n
rightIndex++\n        }\n    }\n    Unit // TODO: Fix KT-31506\n    }\n    }\n    return target\n},"/*\n *
Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is
governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage
kotlin.collections\n\n@OptIn(ExperimentalUnsignedTypes::class)\n@SinceKotlin("1.3")\n@kotlin.js.JsName("
contentDeepHashCodeImpl")\ninternal fun <T> Array<out T>?.contentDeepHashCodeImpl(): Int {\n    if (this ==
null) return 0\n    var result = 1\n    for (element in this) {\n        val elementHash = when {\n            element == null
-> 0\n            isArrayish(element) -> (element.unsafeCast<Array<*>>()).contentDeepHashCodeImpl()\n
            element is UByteArray -> element.contentHashCode()\n            element is UShortArray ->
element.contentHashCode()\n            element is UIntArray -> element.contentHashCode()\n            element is
ULongArray -> element.contentHashCode()\n            else -> element.hashCode()\n        }\n        result = 31 * result + elementHash\n    }\n    return result\n},"/*\n * Copyright 2010-2018 JetBrains s.r.o. and
Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.collections\n\ninternal interface

```

```

EqualityComparator {\n  /**\n   * Subclasses must override to return a value indicating\n   * whether or not two keys or values are equal.\n   */\n  abstract fun equals(value1: Any?, value2: Any?): Boolean\n\n  /**\n   * Subclasses must override to return the hash code of a given key.\n   */\n  abstract fun getHashCode(value: Any?): Int\n\n  object HashCoder : EqualityComparator {\n    override fun equals(value1: Any?, value2: Any?): Boolean = value1 == value2\n    override fun getHashCode(value: Any?): Int = value?.hashCode() ?: 0\n  }\n},"/**\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n * Based on GWT AbstractHashMap\n * Copyright 2008 Google Inc.\n */\npackage kotlin.collections\n\nimport kotlin.collections.MutableMap.MutableEntry\n\n/**\n * Hash table based implementation of the [MutableMap] interface.\n * This implementation makes no guarantees regarding the order of enumeration of [keys], [values] and [entries] collections.\n */\n// Classes that extend HashMap and implement `build()` (freezing) operation\n// have to make sure mutating methods check `checkIsMutable`.\npublic actual open class HashMap<K, V> : AbstractMutableMap<K, V>, MutableMap<K, V> {\n  private inner class EntrySet : AbstractEntrySet<MutableEntry<K, V>, K, V>() {\n    override fun add(element: MutableEntry<K, V>): Boolean = throw UnsupportedOperationException("Add is not supported on entries")\n    override fun clear() {\n      this@HashMap.clear()\n    }\n    override fun containsEntry(element: Map.Entry<K, V>): Boolean = this@HashMap.containsEntry(element)\n    override operator fun iterator(): MutableIterator<MutableEntry<K, V>> = internalMap.iterator()\n    override fun removeEntry(element: Map.Entry<K, V>): Boolean {\n      if (contains(element)) {\n        this@HashMap.remove(element.key)\n      }\n      return true\n    }\n    override val size: Int get() = this@HashMap.size\n  }\n\n  /**\n   * Internal implementation of the map: either string-based or hashcode-based.\n   */\n  private val internalMap: InternalMap<K, V>\n  private val equality: EqualityComparator\n\n  internal constructor(internalMap: InternalMap<K, V>) : super() {\n    this.internalMap = internalMap\n    this.equality = internalMap.equality\n  }\n\n  /**\n   * Constructs an empty [HashMap] instance.\n   */\n  actual constructor() : this(InternalHashCodeMap(EqualityComparator.HashCode))\n\n  /**\n   * Constructs an empty [HashMap] instance.\n   * @param initialCapacity the initial capacity (ignored)\n   * @param loadFactor the load factor (ignored)\n   * @throws IllegalArgumentException if the initial capacity or load factor are negative\n   */\n  actual constructor(initialCapacity: Int, loadFactor: Float) : this() {\n    // This implementation of HashMap has no need of load factors or capacities.\n    require(initialCapacity >= 0) {\n      "Negative initial capacity: $initialCapacity"\n    }\n    require(loadFactor >= 0) {\n      "Non-positive load factor: $loadFactor"\n    }\n  }\n\n  actual constructor(initialCapacity: Int) : this(initialCapacity, 0.0f)\n\n  /**\n   * Constructs an instance of [HashMap] filled with the contents of the specified [original] map.\n   */\n  actual constructor(original: Map<out K, V>) : this() {\n    this.putAll(original)\n  }\n\n  actual override fun clear() {\n    internalMap.clear()\n    // structureChanged(this)\n  }\n\n  actual override fun containsKey(key: K): Boolean = internalMap.containsKey()\n\n  actual override fun containsValue(value: V): Boolean = internalMap.any { equality.equals(it.value, value) }\n\n  private var _entries: MutableSet<MutableMap.MutableEntry<K, V>>? = null\n  actual override val entries: MutableSet<MutableMap.MutableEntry<K, V>>\n    get() {\n      if (_entries == null) {\n        _entries = createEntrySet()\n      }\n      return _entries!!\n    }\n\n  internal open fun createEntrySet(): MutableSet<MutableMap.MutableEntry<K, V>> = EntrySet()\n\n  actual override operator fun get(key: K): V? = internalMap.get(key)\n\n  actual override fun put(key: K, value: V): V? = internalMap.put(key, value)\n\n  actual override fun remove(key: K): V? = internalMap.remove(key)\n\n  actual override val size: Int get() = internalMap.size\n}\n\n/**\n * Constructs the specialized implementation of [HashMap] with [String] keys, which stores the keys as properties of\n * JS object without hashing them.\n */\npublic fun <V> stringMapOf(vararg pairs: Pair<String, V>): HashMap<String, V> {\n  return HashMap<String, V>(InternalStringMap(EqualityComparator.HashCode)).apply { putAll(pairs) }\n}\n"/**\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n * Based on GWT HashSet\n * Copyright 2008 Google Inc.\n */\npackage kotlin.collections\n\n/**\n * The implementation of the

```

```

[MutableSet] interface, backed by a [HashMap] instance.\n *\n// Classes that extend HashSet and implement
`build()` (freezing) operation\n// have to make sure mutating methods check `checkIsMutable`.\n\npublic actual open
class HashSet<E> : AbstractMutableSet<E>, MutableSet<E> {\n\n    internal val map: HashMap<E, Any>\n\n    /**\n     * Constructs a new empty [HashSet].\n     *\n     actual constructor() {\n         map = HashMap<E, Any>()\n     }\n\n     /**\n     * Constructs a new [HashSet] filled with the elements of the specified collection.\n     *\n     actual
constructor(elements: Collection<E>) {\n         map = HashMap<E, Any>(elements.size)\n         addAll(elements)\n     }\n\n     /**\n     * Constructs a new empty [HashSet].\n     *\n     * @param initialCapacity the initial capacity
(ignored)\n     * @param loadFactor the load factor (ignored)\n     *\n     * @throws IllegalArgumentException if
the initial capacity or load factor are negative\n     *\n     actual constructor(initialCapacity: Int, loadFactor: Float)
{\n         map = HashMap<E, Any>(initialCapacity, loadFactor)\n     }\n\n     actual constructor(initialCapacity: Int) :
this(initialCapacity, 0.0f)\n\n     /**\n     * Protected constructor to specify the underlying map. This is used by\n     *
LinkedHashSet.\n     * @param map underlying map to use.\n     *\n     internal constructor(map: HashMap<E,
Any>) {\n         this.map = map\n     }\n\n     actual override fun add(element: E): Boolean {\n         val old =
map.put(element, this)\n         return old == null\n     }\n\n     actual override fun clear() {\n         map.clear()\n     }\n\n
    public override fun clone(): Any {\n\n        return HashSet<E>(this)\n\n    }\n\n     actual override operator fun
contains(element: E): Boolean = map.containsKey(element)\n\n     actual override fun isEmpty(): Boolean =
map.isEmpty()\n\n     actual override fun iterator(): MutableIterator<E> = map.keys.iterator()\n\n     actual override
fun remove(element: E): Boolean = map.remove(element) != null\n\n     actual override val size: Int get() =
map.size\n\n}\n\n/**\n * Creates a new instance of the specialized implementation of [HashSet] with the specified
[String] elements,\n * which elements the keys as properties of JS object without hashing them.\n *\n     actual fun
stringSetOf(vararg elements: String): HashSet<String> {\n         return HashSet(stringMapOf<Any>()).apply {
            addAll(elements)\n        }\n    }\n\n}"/\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n * Based on GWT InternalHashCodeMap\n * Copyright 2008 Google Inc.\n */\n\npackage kotlin.collections\n\nimport kotlin.collections.MutableMap.MutableEntry\nimport
kotlin.collections.AbstractMutableMap.SimpleEntry\n\n/**\n * A simple wrapper around JavaScriptObject to
provide [java.util.Map]-like semantics for any\n * key type.\n * *\n * Implementation notes:\n * *\n * A key's
hashCode is the index in backingMap which should contain that key. Since several keys may\n * have the same
hash, each value in hashCodeMap is actually an array containing all entries whose\n * keys share the same hash.\n *
*\n     internal class InternalHashCodeMap<K, V>(override val equality: EqualityComparator) : InternalMap<K, V>
{\n\n         private var backingMap: dynamic = createJsMap()\n         override var size: Int = 0\n         private set\n\n         override fun put(key: K, value: V): V? {\n             val hashCode = equality.getHashCode(key)\n             val chainOrEntry
= getChainOrEntryOrNull(hashCode)\n             if (chainOrEntry == null) {\n                 // This is a new chain, put it to the
map.\n                 backingMap[hashCode] = SimpleEntry(key, value)\n             } else {\n                 if (chainOrEntry !is
Array<*>) {\n                     // It is an entry\n                     val entry: SimpleEntry<K, V> = chainOrEntry\n                     if
(equality.equals(entry.key, key)) {\n                         return entry.setValue(value)\n                     } else {\n
backingMap[hashCode] = arrayOf(entry, SimpleEntry(key, value))\n                     size++\n                     return null\n
}\n                 } else {\n                     // Chain already exists, perhaps key also exists.\n                     val chain:
Array<MutableEntry<K, V>> = chainOrEntry\n                     val entry = chain.findEntryInChain(key)\n                     if
(entry != null) {\n                         return entry.setValue(value)\n                     }\n                 }\n\n                 chain.asDynamic().push(SimpleEntry(key, value))\n                 }\n                 size++\n                 structureChanged(host)\n                 return null\n             }\n\n             override fun remove(key: K): V? {\n                 val hashCode = equality.getHashCode(key)\n                 val chainOrEntry = getChainOrEntryOrNull(hashCode) ?: return null\n                 if (chainOrEntry !is Array<*>) {\n                     val entry: MutableEntry<K, V> = chainOrEntry\n                     if (equality.equals(entry.key, key)) {\n
jsDeleteProperty(backingMap, hashCode)\n                     size--\n                     return entry.value\n                 } else {\n
return null\n                 }\n                 } else {\n                     val chain: Array<MutableEntry<K, V>> = chainOrEntry\n                     for
(index in chain.indices) {\n                         val entry = chain[index]\n                         if (equality.equals(key, entry.key)) {\n
if (chain.size == 1) {\n                             chain.asDynamic().length = 0\n                             // remove the whole

```



```

guaranteed to be still correct.\n// * This is to optimize for the common scenario where the values are not modified
during\n// * iterations where the entries are never stale.\n// */\n// private var valueMod: Int = 0\n\n override
operator fun contains(key: K): Boolean {\n    if (key !is String) return false\n    return backingMap[key] !==
undefined\n }\n\n override operator fun get(key: K): V? {\n    if (key !is String) return null\n    val value =
backingMap[key]\n    return if (value !== undefined) value.unsafeCast<V>() else null\n }\n\n\n override fun
put(key: K, value: V): V? {\n    require(key is String)\n    val oldValue = backingMap[key]\n
backingMap[key] = value\n\n    if (oldValue === undefined) {\n        size++\n//
structureChanged(host)\n        return null\n    } else {\n//        valueMod++\n        return
oldValue.unsafeCast<V>()\n    }\n }\n\n\n override fun remove(key: K): V? {\n    if (key !is String) return
null\n    val value = backingMap[key]\n    if (value !== undefined) {\n        jsDeleteProperty(backingMap,
key)\n        size--\n//        structureChanged(host)\n        return value.unsafeCast<V>()\n    } else {\n//
valueMod++\n        return null\n    }\n }\n\n\n\n override fun clear() {\n    backingMap = createJsMap()\n
size = 0\n }\n\n\n\n override fun iterator(): MutableIterator<MutableEntry<K, V>> {\n    return object :
MutableIterator<MutableEntry<K, V>> {\n        private val keys: Array<String> =
js("Object").keys(backingMap)\n        private val iterator = keys.iterator()\n        private var lastKey: String? =
null\n\n        override fun hasNext(): Boolean = iterator.hasNext()\n\n        override fun next():
MutableEntry<K, V> {\n            val key = iterator.next()\n            lastKey = key\n
@Suppress("UNCHECKED_CAST")\n            return newMapEntry(key as K)\n        }\n\n        override
fun remove() {\n            @Suppress("UNCHECKED_CAST")\n            this@InternalStringMap.remove(checkNotNull(lastKey) as K)\n        }\n    }\n }\n\n\n private fun
newMapEntry(key: K): MutableEntry<K, V> = object : MutableEntry<K, V> {\n    override val key: K get() =
key\n    override val value: V get() = this@InternalStringMap[key].unsafeCast<V>()\n\n    override fun
setValue(newValue: V): V = this@InternalStringMap.put(key, newValue).unsafeCast<V>()\n\n    override fun
hashCode(): Int = AbstractMap.entryHashCode(this)\n\n    override fun toString(): String =
AbstractMap.entryToString(this)\n\n    override fun equals(other: Any?): Boolean = AbstractMap.entryEquals(this,
other)\n }\n}\n\n", "/*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n *
Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*\n */\n\n * Based on GWT LinkedHashMap\n * Copyright 2008 Google Inc.\n */\n\npackage
kotlin.collections\n\nimport kotlin.collections.MutableMap.MutableEntry\n\n/**\n * Hash table based
implementation of the [MutableMap] interface, which additionally preserves the insertion order\n * of entries during
the iteration.\n * The insertion order is preserved by maintaining a doubly-linked list of all of its entries.\n
*\n */\n\npublic actual open class LinkedHashMap<K, V> : HashMap<K, V>, MutableMap<K, V> {\n\n    /**\n     * The
entry we use includes next/prev pointers for a doubly-linked circular\n     * list with a head node. This reduces the
special cases we have to deal with\n     * in the list operations.\n     * Note that we duplicate the key from the
underlying hash map so we can find\n     * the eldest entry. The alternative would have been to modify HashMap so
more\n     * of the code was directly usable here, but this would have added some\n     * overhead to HashMap, or to
reimplement most of the HashMap code here with\n     * small modifications. Paying a small storage cost only if
you use\n     * LinkedHashMap and minimizing code size seemed like a better tradeoff\n     */\n\n    private inner class
ChainEntry<K, V>(key: K, value: V) : AbstractMutableMap.SimpleEntry<K, V>(key, value) {\n        internal var
next: ChainEntry<K, V>? = null\n        internal var prev: ChainEntry<K, V>? = null\n\n        override fun
setValue(newValue: V): V {\n            this@LinkedHashMap.checkIsMutable()\n            return
super.setValue(newValue)\n        }\n    }\n\n    private inner class EntrySet : AbstractEntrySet<MutableEntry<K,
V>, K, V>() {\n        private inner class EntryIterator : MutableIterator<MutableEntry<K, V>> {\n            // The
last entry that was returned from this iterator.\n            private var last: ChainEntry<K, V>? = null\n            // The
next entry to return from this iterator.\n            private var next: ChainEntry<K, V>? = null\n\n            init {\n
                next = head\n//                recordLastKnownStructure(map, this)\n            }\n\n            override fun hasNext():
Boolean {\n                return next !== null\n            }\n\n            override fun next(): MutableEntry<K, V> {\n//
                checkStructuralChange(map, this)\n                if (!hasNext()) throw NoSuchElementException()\n\n                val

```

```

current = next!!\n          last = current\n          next = current.next.takeIf { it !== head }\n          return
current\n          }\n          override fun remove() {\n          check(last != null)\n
this@EntrySet.checkIsMutable()\n//          checkStructuralChange(map, this)\n          last!!.remove()\n
map.remove(last!!.key)\n//          recordLastKnownStructure(map, this)\n          last = null\n          }\n
}\n          override fun add(element: MutableEntry<K, V>): Boolean = throw
UnsupportedOperationException("Add is not supported on entries")\n          override fun clear() {\n
this@LinkedHashMap.clear()\n          }\n          override fun containsEntry(element: Map.Entry<K, V>): Boolean =
this@LinkedHashMap.containsEntry(element)\n          override operator fun iterator():
MutableIterator<MutableEntry<K, V>> = EntryIterator()\n          override fun removeEntry(element: Map.Entry<K,
V>): Boolean {\n          checkIsMutable()\n          if (contains(element)) {\n
this@LinkedHashMap.remove(element.key)\n          return true\n          }\n          return false\n          }\n
          override val size: Int get() = this@LinkedHashMap.size\n          override fun checkIsMutable(): Unit =
this@LinkedHashMap.checkIsMutable()\n          }\n          /*\n          * The head of the insert order chain, which is a doubly-
linked circular\n          * list.\n          * \n          * The most recently inserted node is at the end of the chain, ie.\n          * chain.prev.\n          * \n          * private var head: ChainEntry<K, V>? = null\n          * \n          * Add this node to the end of the chain.\n          * \n          * private fun ChainEntry<K, V>.addToEnd() {\n          // This entry is not in the list.\n          check(next == null && prev
== null)\n          val _head = head\n          if (_head == null) {\n          head = this\n          next = this\n          prev =
this\n          } else {\n          // Chain is valid.\n          val _tail = checkNotNull(_head.prev)\n          // Update me.\n          prev = _tail\n          next = _head\n          // Update my new siblings: current head and old tail\n          _head.prev = this\n          _tail.next = this\n          }\n          }\n          /*\n          * Remove this node from the chain it is a part
of.\n          * \n          * private fun ChainEntry<K, V>.remove() {\n          if (this.next === this) {\n          // if this is single
element, remove head\n          head = null\n          } else {\n          if (head === this) {\n          // if this is first
element, move head to next\n          head = next\n          }\n          next!!.prev = prev\n          prev!!.next =
next\n          }\n          next = null\n          prev = null\n          }\n          }\n          /*\n          * The hashmap that keeps track of our entries and
the chain. Note that we\n          * duplicate the key here to eliminate changes to HashMap and minimize the\n          * code
here, at the expense of additional space.\n          * \n          * private val map: HashMap<K, ChainEntry<K, V>>\n          private
var isReadOnly: Boolean = false\n          /*\n          * Constructs an empty [LinkedHashMap] instance.\n          * \n          * actual
constructor() : super() {\n          map = HashMap<K, ChainEntry<K, V>>()\n          }\n          internal
constructor(backingMap: HashMap<K, Any>) : super() {\n          @Suppress("UNCHECKED_CAST") // expected
to work due to erasure\n          map = backingMap as HashMap<K, ChainEntry<K, V>>\n          }\n          /*\n          *
Constructs an empty [LinkedHashMap] instance.\n          * \n          * @param initialCapacity the initial capacity
(ignored)\n          * @param loadFactor the load factor (ignored)\n          * \n          * @throws IllegalArgumentException if
the initial capacity or load factor are negative\n          * \n          * actual constructor(initialCapacity: Int, loadFactor: Float) :
super(initialCapacity, loadFactor) {\n          map = HashMap<K, ChainEntry<K, V>>()\n          }\n          actual
constructor(initialCapacity: Int) : this(initialCapacity, 0.0f)\n          /*\n          * Constructs an instance of
[LinkedHashMap] filled with the contents of the specified [original] map.\n          * \n          * actual constructor(original:
Map<out K, V>) {\n          map = HashMap<K, ChainEntry<K, V>>()\n          this.putAll(original)\n          }\n          }
          @PublishedApi\n          internal fun build(): Map<K, V> {\n          checkIsMutable()\n          isReadOnly = true\n          return this\n          }\n          actual override fun clear() {\n          checkIsMutable()\n          map.clear()\n          head = null\n          }\n          }\n          // override fun clone(): Any {\n          // return LinkedHashMap(this)\n          // }\n          }\n          actual override fun
containsKey(key: K): Boolean = map.containsKey(key)\n          actual override fun containsValue(value: V): Boolean
{\n          var node: ChainEntry<K, V> = head ?: return false\n          do {\n          if (node.value == value) {\n          return true\n          }\n          node = node.next!!\n          } while (node !== head)\n          return false\n          }\n          }\n          internal override fun createEntrySet(): MutableSet<MutableMap.MutableEntry<K, V>> = EntrySet()\n          actual
override operator fun get(key: K): V? = map.get(key)?.value\n          actual override fun put(key: K, value: V): V? {\n          checkIsMutable()\n          val old = map.get(key)\n          if (old == null) {\n          val newEntry =
ChainEntry(key, value)\n          map.put(key, newEntry)\n          newEntry.addToEnd()\n          return null\n          } else {\n          return old.setValue(value)\n          }\n          }\n          }\n          actual override fun remove(key: K): V? {\n

```

```

checkIsMutable()\n    val entry = map.remove(key)\n    if (entry != null) {\n        entry.remove()\n    }\n    return entry.value\n}\nreturn null\n}\n\nactual override val size: Int get() = map.size\n\ninternal\noverride fun checkIsMutable() {\n    if (isReadOnly) throw UnsupportedOperationException()\n}\n}\n\n/**\n * Constructs the specialized implementation of [LinkedHashMap] with [String] keys, which stores the keys as\n * properties of\n * JS object without hashing them.\n */\npublic fun <V> linkedStringMapOf(vararg pairs:\n * Pair<String, V>): LinkedHashMap<String, V> {\n    return LinkedHashMap<String,\n * V>(stringMapOf<Any>()).apply { putAll(pairs) }\n}\n}\n\n"/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin\n * Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be\n * found in the license/LICENSE.txt file.\n */\n\n/**\n * Based on GWT LinkedHashMap\n * Copyright 2008 Google\n * Inc.\n */\n\npackage kotlin.collections\n\n/**\n * The implementation of the [MutableSet] interface, backed by a\n * [LinkedHashMap] instance.\n */\n * This implementation preserves the insertion order of elements during the\n * iteration.\n */\npublic actual open class LinkedHashMap<E> : HashSet<E>, MutableSet<E> {\n\n    internal\n    constructor(map: LinkedHashMap<E, Any>) : super(map)\n\n    /**\n     * Constructs a new empty\n     * [LinkedHashSet].\n     */\n    actual constructor() : super(LinkedHashMap<E, Any>())\n\n    /**\n     * Constructs a\n     * new [LinkedHashSet] filled with the elements of the specified collection.\n     */\n    actual constructor(elements:\n * Collection<E>) : super(LinkedHashMap<E, Any>()) {\n        addAll(elements)\n    }\n\n    /**\n     * Constructs a\n     * new empty [LinkedHashSet].\n     */\n    * @param initialCapacity the initial capacity (ignored)\n     * @param\n * loadFactor the load factor (ignored)\n     */\n    * @throws IllegalArgumentException if the initial capacity or\n * load factor are negative\n     */\n    actual constructor(initialCapacity: Int, loadFactor: Float) :\n * super(LinkedHashMap<E, Any>(initialCapacity, loadFactor))\n\n    actual constructor(initialCapacity: Int) :\n * this(initialCapacity, 0.0f)\n\n    @PublishedApi\n    internal fun build(): Set<E> {\n        (map as\n * LinkedHashMap<E, Any>).build()\n        return this\n    }\n\n    internal override fun checkIsMutable(): Unit =\n * map.checkIsMutable()\n\n    public override fun clone(): Any {\n        return LinkedHashMap(this)\n    }\n}\n\n}\n\n/**\n * Creates a new instance of the specialized implementation of [LinkedHashSet] with the specified\n * [String] elements,\n * which elements the keys as properties of JS object without hashing them.\n */\npublic fun\n * linkedStringSetOf(vararg elements: String): LinkedHashMap<String> {\n    return\n * LinkedHashMap(linkedStringMapOf<Any>()).apply { addAll(elements) }\n}\n}\n\n"/*\n * Copyright 2010-2020\n * JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the\n * Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin\n\nimport\n * kotlin.contracts.*\n\n@DeprecatedSinceKotlin(warningSince = \"1.6\")\n@Deprecated(\"Synchronization on any\n * object is not supported in Kotlin/JS\",  
 * ReplaceWith(\"run(block)\"))\n@kotlin.internal.InlineOnly\n@Suppress(\"UNUSED_PARAMETER\")\npublic\n * inline fun <R> synchronized(lock: Any, block: () -> R): R {\n    contract {\n        callsInPlace(block,\n * InvocationKind.EXACTLY_ONCE)\n    }\n    return block()\n}\n}\n\n"/*\n * Copyright 2010-2018 JetBrains s.r.o. and\n * Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that\n * can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.io\n\ninternal abstract class BaseOutput {\n\n    open fun println() {\n        print(\"\\n\")\n    }\n\n    open fun println(message: Any?) {\n        print(message)\n        println()\n    }\n\n    abstract fun print(message: Any?)\n\n    open fun flush() {\n    }\n}\n\n/** JsName used to make the\n * declaration available outside of module to test it\n */\n@JsName(\"NodeJsOutput\")\ninternal class NodeJsOutput(val\n * outputStream: dynamic) : BaseOutput() {\n    override fun print(message: Any?) {\n        // TODO: Using local\n * variable because of bug in block decomposition lowering in IR backend\n        val messageString =\n * String(message)\n        outputStream.write(messageString)\n    }\n}\n\n/** JsName used to make the declaration\n * available outside of module to test it\n */\n@JsName(\"OutputToConsoleLog\")\ninternal class OutputToConsoleLog\n * : BaseOutput() {\n    override fun print(message: Any?) {\n        console.log(message)\n    }\n\n    override fun\n * println(message: Any?) {\n        console.log(message)\n    }\n\n    override fun println() {\n        console.log(\"\")\n    }\n}\n\n/** JsName used to make the declaration available outside of module to test it and use at try.kotl.in\n */\n@JsName(\"BufferedOutput\")\ninternal open class BufferedOutput : BaseOutput() {\n    var buffer = \"\"\n\n    override fun print(message: Any?) {\n        buffer += String(message)\n    }\n\n    override fun flush() {\n        buffer

```

```

= "\n } } \n\n** JsName used to make the declaration available outside of module to test it
*\n@JsName("BufferedOutputToConsoleLog")\ninternal class BufferedOutputToConsoleLog : BufferedOutput()
{\n  override fun print(message: Any?) {\n    var s = String(message)\n    val i = s.nativeLastIndexOf("\n",
0)\n    if (i >= 0) {\n      buffer += s.substring(0, i)\n      flush()\n      s = s.substring(i + 1)\n    }\n    buffer += s\n  }\n  override fun flush() {\n    console.log(buffer)\n    buffer = "\n } } \n\n** JsName
used to make the declaration available outside of module to test it and use at try.kotl.in
*\n@JsName("output")\ninternal var output = run {\n  val isNode: Boolean = js("typeof process !== 'undefined'
&& process.versions && !!process.versions.node")\n  if (isNode) NodeJsOutput(js("process.stdout")) else
BufferedOutputToConsoleLog()\n}\n\n@kotlin.internal.InlineOnly\nprivate inline fun String(value: Any?): String =
js("String")(value)\n\n** Prints the line separator to the standard output stream. *\npublic actual fun println() {\n
output.println()\n}\n\n** Prints the given [message] and the line separator to the standard output stream. *\npublic
actual fun println(message: Any?) {\n  output.println(message)\n}\n\n** Prints the given [message] to the standard
output stream. *\npublic actual fun print(message: Any?) {\n
output.print(message)\n}\n\n@SinceKotlin("1.6")\npublic actual fun readln(): String = throw
UnsupportedOperationException("readln is not supported in Kotlin/JS")\n\n@SinceKotlin("1.6")\npublic actual
fun readlnOrNull(): String? = throw UnsupportedOperationException("readlnOrNull is not supported in
Kotlin/JS"), /*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use
of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*\n\npackage kotlin.coroutines\n\nimport kotlin.coroutines.intrinsics.CoroutineSingletons.*\nimport
kotlin.coroutines.intrinsics.COROUTINE_SUSPENDED\n\n@PublishedApi\n@SinceKotlin("1.3")\ninternal
actual class SafeContinuation<in T>\ninternal actual constructor(\n  private val delegate: Continuation<T>,\n
initialResult: Any?)\n) : Continuation<T> {\n  @PublishedApi\n  internal actual constructor(delegate:
Continuation<T>) : this(delegate, UNDECIDED)\n\n  public actual override val context: CoroutineContext\n
get() = delegate.context\n\n  private var result: Any? = initialResult\n\n  public actual override fun
resumeWith(result: Result<T>) {\n    val cur = this.result\n    when {\n      cur === UNDECIDED -> {\n
this.result = result.value\n    }\n      cur === COROUTINE_SUSPENDED -> {\n      this.result =
RESUMED\n      delegate.resumeWith(result)\n    }\n      else -> throw
IllegalStateException("Already resumed")\n    }\n  }\n\n  @PublishedApi\n  internal actual fun
getOrThrow(): Any? {\n    if (result === UNDECIDED) {\n      result = COROUTINE_SUSPENDED\n
return COROUTINE_SUSPENDED\n    }\n    val result = this.result\n    return when {\n      result ===
RESUMED -> COROUTINE_SUSPENDED // already called continuation, indicate COROUTINE_SUSPENDED
upstream\n      result is Result.Failure -> throw result.exception\n      else -> result // either
COROUTINE_SUSPENDED or data\n    }\n  }\n}\n\n", /*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin
Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.\n *\n\npackage
kotlin.coroutines.cancellation\n\n@SinceKotlin("1.4")\npublic actual open class CancellationException :
IllegalStateException {\n  actual constructor() : super()\n  actual constructor(message: String?) : super(message)\n
  constructor(message: String?, cause: Throwable?) : super(message, cause)\n  constructor(cause: Throwable?) :
super(cause)\n}\n\n", /*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n *
Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*\n\npackage kotlin.coroutines.js.internal\n\nimport kotlin.coroutines.Continuation\nimport
kotlin.coroutines.EmptyCoroutineContext\n\n@PublishedApi\n@SinceKotlin("1.3")\ninternal val
EmptyContinuation = Continuation<Any?>(EmptyCoroutineContext) { result ->\n  result.getOrThrow()\n}\n\n", /*\n *
Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code
is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n *\n\npackage
kotlin.js\n\n**\n * Exposes the [Date API](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Date) to Kotlin.\n
*\n\n@Suppress("NOT_DOCUMENTED")\npublic external class Date() {\n  public constructor(milliseconds:

```

```

Number)\n\n public constructor(dateString: String)\n\n public constructor(year: Int, month: Int)\n\n public
constructor(year: Int, month: Int, day: Int)\n\n public constructor(year: Int, month: Int, day: Int, hour: Int)\n\n
public constructor(year: Int, month: Int, day: Int, hour: Int, minute: Int)\n\n public constructor(year: Int, month:
Int, day: Int, hour: Int, minute: Int, second: Int)\n\n public constructor(year: Int, month: Int, day: Int, hour: Int,
minute: Int, second: Int, millisecond: Number)\n\n public fun getDate(): Int\n\n public fun getDay(): Int\n\n
public fun getFullYear(): Int\n\n public fun getHours(): Int\n\n public fun getMilliseconds(): Int\n\n public fun
getMinutes(): Int\n\n public fun getMonth(): Int\n\n public fun getSeconds(): Int\n\n public fun getTime():
Double\n\n public fun getTimezoneOffset(): Int\n\n public fun getUTCDate(): Int\n\n public fun
getUTCDay(): Int\n\n public fun getUTCFullYear(): Int\n\n public fun getUTCHours(): Int\n\n public fun
getUTCMilliseconds(): Int\n\n public fun getUTCMinutes(): Int\n\n public fun getUTCMonth(): Int\n\n public
fun getUTCSeconds(): Int\n\n public fun toDateString(): String\n\n public fun toISOString(): String\n\n public
fun toJSON(): Json\n\n public fun toLocaleDateString(locales: Array<String> = definedExternally, options:
LocaleOptions = definedExternally): String\n\n public fun toLocaleDateString(locales: String, options:
LocaleOptions = definedExternally): String\n\n public fun toLocaleString(locales: Array<String> =
definedExternally, options: LocaleOptions = definedExternally): String\n\n public fun toLocaleString(locales:
String, options: LocaleOptions = definedExternally): String\n\n public fun toLocaleString(locales:
Array<String> = definedExternally, options: LocaleOptions = definedExternally): String\n\n public fun
toLocaleString(locales: String, options: LocaleOptions = definedExternally): String\n\n public fun
toTimeString(): String\n\n public fun toUTCString(): String\n\n public companion object {\n public fun
now(): Double\n\n public fun parse(dateString: String): Double\n\n public fun UTC(year: Int, month: Int):
Double\n\n public fun UTC(year: Int, month: Int, day: Int): Double\n\n public fun UTC(year: Int, month:
Int, day: Int, hour: Int): Double\n\n public fun UTC(year: Int, month: Int, day: Int, hour: Int, minute: Int):
Double\n\n public fun UTC(year: Int, month: Int, day: Int, hour: Int, minute: Int, second: Int): Double\n\n
public fun UTC(year: Int, month: Int, day: Int, hour: Int, minute: Int, second: Int, millisecond: Number): Double\n
}\n\n public interface LocaleOptions {\n public var localeMatcher: String?\n\n public var timeZone:
String?\n\n public var hour12: Boolean?\n\n public var formatMatcher: String?\n\n public var weekday:
String?\n\n public var era: String?\n\n public var year: String?\n\n public var month: String?\n\n
public var day: String?\n\n public var hour: String?\n\n public var minute: String?\n\n public var
second: String?\n\n public var timeZoneName: String?\n }\n}\n\npublic inline fun dateLocaleOptions(init:
Date.LocaleOptions.() -> Unit): Date.LocaleOptions {\n val result = js("new
Object()\").unsafeCast<Date.LocaleOptions>()\n init(result)\n return result\n},"/*\n * Copyright 2010-2020
JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.dom\n\nimport
org.w3c.dom.Document\n\nimport org.w3c.dom.Element\n\nimport
kotlin.internal.LowPriorityInOverloadResolution\n\nimport kotlinx.dom.appendElement as
newAppendElement\n\nimport kotlinx.dom.createElement as newCreateElement\n\n/**\n * Creates a new element
with the specified [name].\n * The element is initialized with the specified [init] function.\n
*\n */\n\n@LowPriorityInOverloadResolution\n@Deprecated(\n message = "This API is moved to another package,
use 'kotlinx.dom.createElement' instead.",\n replaceWith = ReplaceWith("this.createElement(name, init)",
"kotlinx.dom.createElement")\n)\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.6")\npublic
inline fun Document.createElement(name: String, noinline init: Element.() -> Unit): Element =
this.newCreateElement(name, init)\n\n/**\n * Appends a newly created element with the specified [name] to this
element.\n * The element is initialized with the specified [init] function.\n
*\n */\n\n@LowPriorityInOverloadResolution\n@Deprecated(\n message = "This API is moved to another package,
use 'kotlinx.dom.appendElement' instead.",\n replaceWith = ReplaceWith("this.appendElement(name, init)",
"kotlinx.dom.appendElement")\n)\n@DeprecatedSinceKotlin(warningSince = "1.4", errorSince = "1.6")\npublic
inline fun Element.appendElement(name: String, noinline init: Element.() -> Unit): Element =
this.newAppendElement(name, init)\n\n",/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming

```

```

Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\npackage kotlin.dom\n\nimport org.w3c.dom.Element\nimport
kotlin.internal.LowPriorityInOverloadResolution\nimport kotlin.dom.addClass as newAddClass\nimport
kotlin.dom.hasClass as newHasClass\nimport kotlin.dom.removeClass as newRemoveClass\n\n/** Returns true if
the element has the given CSS class style in its 'class' attribute
*/\n\n@LowPriorityInOverloadResolution\n@Deprecated(\n    message = \"This API is moved to another package,
use 'kotlin.dom.hasClass' instead.\",\n    replaceWith = ReplaceWith(\"this.hasClass(cssClass)\",
\"kotlin.dom.hasClass\")\n)\n\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince = \"1.6\")\ninline fun
Element.hasClass(cssClass: String): Boolean = this.newHasClass(cssClass)\n\n/**\n * Adds CSS class to element.
Has no effect if all specified classes are already in class attribute of the element\n * \n * @return true if at least one
class has been added\n */\n\n@LowPriorityInOverloadResolution\n@Deprecated(\n    message = \"This API is moved
to another package, use 'kotlin.dom.addClass' instead.\",\n    replaceWith =
ReplaceWith(\"this.addClass(cssClasses)\", \"kotlin.dom.addClass\")\n)\n\n@DeprecatedSinceKotlin(warningSince
= \"1.4\", errorSince = \"1.6\")\ninline fun Element.addClass(vararg cssClasses: String): Boolean =
this.newAddClass(*cssClasses)\n\n/**\n * Removes all [cssClasses] from element. Has no effect if all specified
classes are missing in class attribute of the element\n * \n * @return true if at least one class has been removed\n
*/\n\n@LowPriorityInOverloadResolution\n@Deprecated(\n    message = \"This API is moved to another package,
use 'kotlin.dom.removeClass' instead.\",\n    replaceWith = ReplaceWith(\"this.removeClass(cssClasses)\",
\"kotlin.dom.removeClass\")\n)\n\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince = \"1.6\")\ninline
fun Element.removeClass(vararg cssClasses: String): Boolean = this.newRemoveClass(*cssClasses)\n\n/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is
governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage
kotlin.dom\n\nimport org.w3c.dom.Element\nimport org.w3c.dom.Node\nimport
kotlin.internal.LowPriorityInOverloadResolution\nimport kotlin.dom.isElement as newIsElement\nimport
kotlin.dom.isText as newIsText\n\n/**\n * Gets a value indicating whether this node is a TEXT_NODE or a
CDATA_SECTION_NODE.\n */\n\n@LowPriorityInOverloadResolution\n@Deprecated(\n    message = \"This API
is moved to another package, use 'kotlin.dom.isText' instead.\",\n    replaceWith = ReplaceWith(\"this.isText\",
\"kotlin.dom.isText\")\n)\n\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince = \"1.6\")\npublic val
Node.isText: Boolean\n    inline get() = this.newIsText\n\n/**\n * Gets a value indicating whether this node is an
[Element].\n */\n\n@LowPriorityInOverloadResolution\n@Deprecated(\n    message = \"This API is moved to
another package, use 'kotlin.dom.isElement' instead.\",\n    replaceWith = ReplaceWith(\"this.isElement\",
\"kotlin.dom.isElement\")\n)\n\n@DeprecatedSinceKotlin(warningSince = \"1.4\", errorSince = \"1.6\")\npublic val
Node.isElement: Boolean\n    inline get() = this.newIsElement\n\n/*\n * Copyright 2010-2018 JetBrains s.r.o. and
Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n */\n\npackage org.w3c.dom.events\n\npublic fun
EventListener(handler: (Event) -> Unit): EventListener = EventListenerHandler(handler)\n\nprivate class
EventListenerHandler(private val handler: (Event) -> Unit) : EventListener {\n    public override fun
handleEvent(event: Event) {\n        handler(event)\n    }\n\n    public override fun toString(): String =
\"EventListenerHandler($handler)\"\n\n}\n\n/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\npackage org.w3c.dom\n\npublic external interface ItemArrayLike<out T> {\n
val length: Int\n    fun item(index: Int): T?\n\n}\n\n/**\n * Returns the view of this `ItemArrayLike<T>` collection as
`List<T>`\n */\n\npublic fun <T> ItemArrayLike<T>.asList(): List<T> = object : AbstractList<T>() {\n    override val
size: Int get() = this@asList.length\n\n    override fun get(index: Int): T = when (index) {\n        in 0..lastIndex ->
this@asList.item(index).unsafeCast<T>()\n        else -> throw IndexOutOfBoundsException(\"index $index is not in
range [0..$lastIndex]\")\n    }\n\n}\n\n/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\npackage kotlin.dom\n\nimport org.w3c.dom.Element\nimport

```

```

org.w3c.dom.Node\nimport kotlin.internal.LowPriorityInOverloadResolution\nimport kotlin.dom.appendText as
newAppendText\nimport kotlin.dom.clear as newClear\n\n/** Removes all the children from this node.
*\n@\n@LowPriorityInOverloadResolution\n@Deprecated(\n    message = `\"This API is moved to another package,
use 'kotlinx.dom.clear' instead.`,\n    replaceWith = ReplaceWith(`\"this.clear()`),
`\"kotlinx.dom.clear()`)\n)\n@DeprecatedSinceKotlin(warningSince = `\"1.4\"`, errorSince = `\"1.6\"`)\npublic inline fun
Node.clear() = this.newClear()\n\n/**\n * Creates text node and append it to the element.\n * @return this
element\n *@\n@LowPriorityInOverloadResolution\n@Deprecated(\n    message = `\"This API is moved to another
package, use 'kotlinx.dom.appendText' instead.`,\n    replaceWith = ReplaceWith(`\"this.appendText(text)`),
`\"kotlinx.dom.appendText()`)\n)\n@DeprecatedSinceKotlin(warningSince = `\"1.4\"`, errorSince = `\"1.6\"`)\ninline fun
Element.appendText(text: String): Element = this.newAppendText(text)\n\n\", /*\n * Copyright 2010-2018 JetBrains
s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0
license that can be found in the license/LICENSE.txt file.\n */\npackage kotlin.js\n\n/**\n * Reinterprets this value
as a value of the [dynamic type](/docs/reference/dynamic-type.html).\n *@\n@kotlin.internal.InlineOnly\npublic
inline fun Any?.asDynamic(): dynamic = this\n\n/**\n * Reinterprets this value as a value of the specified type [T]
without any actual type checking.\n *@\n@kotlin.internal.InlineOnly\npublic inline fun <T> Any?.unsafeCast():
@kotlin.internal.NoInfer T = this.asDynamic()\n\n/**\n * Reinterprets this `dynamic` value as a value of the
specified type [T] without any actual type checking.\n
*\n@\n@kotlin.internal.DynamicExtension\n@JsName(`\"unsafeCastDynamic\"`)\n@kotlin.internal.InlineOnly\npublic
inline fun <T> dynamic.unsafeCast(): @kotlin.internal.NoInfer T = this\n\n/**\n * Allows to iterate this `dynamic`
object in the following cases:\n * - when it has an `iterator` function,\n * - when it is an array\n * - when it is an
instance of [kotlin.collections.Iterable]\n *@\n@kotlin.internal.DynamicExtension\npublic operator fun
dynamic.iterator(): Iterator<dynamic> {\n    val r: Any? = this\n\n    return when {\n        this[\"iterator\"] != null -
>\n            this[\"iterator\"]()\n        isArrayish(r) ->\n            r.unsafeCast<Array<*>>().iterator()\n        else ->\n            (r as Iterable<*>).iterator()\n    }\n}\n\n\", /*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\n// a package is omitted to get declarations directly under the
module\n\n@JsName(`\"throwNPE\"`)\ninternal fun throwNPE(message: String) {\n    throw
NullPointerException(message)\n}\n\n@JsName(`\"throwCCE\"`)\ninternal fun throwCCE() {\n    throw
ClassCastException(`\"Illegal cast\"`)\n}\n\n@JsName(`\"throwISE\"`)\ninternal fun throwISE(message: String) {\n
throw IllegalStateException(message)\n}\n\n@JsName(`\"throwUPAE\"`)\ninternal fun throwUPAE(propertyName:
String) {\n    throw UninitializedPropertyAccessException(`\"lateinit property ${propertyName} has not been
initialized\"`)\n}\n\n\", /*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n
*\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*\n */\npackage kotlin.collections\n\n/**\n * Groups elements from the [Grouping] source by key and counts elements
in each group.\n * @return a [Map] associating the key of each group with the count of elements in the group.\n
*\n * @sample samples.collections.Grouping.groupingByEachCount\n *@\n@SinceKotlin(`\"1.1\"`)\npublic actual fun
<T, K> Grouping<T, K>.eachCount(): Map<K, Int> =\n    fold(0) { acc, _ -> acc + 1 }\n\n/**\n * Groups
elements from the [Grouping] source by key and sums values provided by the [valueSelector] function for elements
in each group.\n * @return a [Map] associating the key of each group with the count of element in the group.\n
*\n */\n@SinceKotlin(`\"1.1\"`)\npublic inline fun <T, K> Grouping<T, K>.eachSumOf(valueSelector: (T) -> Int):
Map<K, Int> =\n    fold(0) { acc, e -> acc + valueSelector(e) }\n\n\", /*\n * Copyright 2010-2018 JetBrains s.r.o.
and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license
that can be found in the license/LICENSE.txt file.\n
*\n */\n@file:kotlin.jvm.JvmName(`\"GroupingKt\"`)\n@file:kotlin.jvm.JvmMultifileClass\n\npackage
kotlin.collections\n\n/**\n * Represents a source of elements with a [keyOf] function, which can be applied to each
element to get its key.\n * @A [Grouping] structure serves as an intermediate step in group-and-fold operations:\n
* they group elements by their keys and then fold each group with some aggregating operation.\n * @It is created
by attaching `keySelector: (T) -> K` function to a source of elements.\n * @To get an instance of [Grouping] use one

```

```

of `groupBy` extension functions:\n * - [Iterable.groupBy]\n * - [Sequence.groupBy]\n * -
[Array.groupBy]\n * - [CharSequence.groupBy]\n * \n * For the list of group-and-fold operations available,
see the [extension functions](#extension-functions) for `Grouping`.\n * \n * @SinceKotlin("1.1")\n * public interface
Grouping<T, out K> {\n *     /** Returns an [Iterator] over the elements of the source of this grouping. *\n *     fun
sourceIterator(): Iterator<T>\n *     /** Extracts the key of an [element]. *\n *     fun keyOf(element: T): K\n * }\n * \n * Groups elements from the [Grouping] source by key and applies [operation] to the elements of each group
sequentially,\n * passing the previously accumulated value and the current element as arguments, and stores the
results in a new map.\n * \n * The key for each element is provided by the [Grouping.keyOf] function.\n * \n *
@param operation function is invoked on each element with the following parameters:\n *     - `key`: the key of the
group this element belongs to;\n *     - `accumulator`: the current value of the accumulator of the group, can be `null`
if it's the first `element` encountered in the group;\n *     - `element`: the element from the source being aggregated;\n
*     - `first`: indicates whether it's the first `element` encountered in the group.\n * \n * @return a [Map] associating
the key of each group with the result of aggregation of the group elements.\n * \n * @sample
samples.collections.Grouping.aggregateByRadix\n * \n * @SinceKotlin("1.1")\n * public inline fun <T, K, R>
Grouping<T, K>.aggregate(\n *     operation: (key: K, accumulator: R?, element: T, first: Boolean) -> R\n * ): Map<K,
R> {\n *     return aggregateTo(mutableMapOf<K, R>(), operation)\n * }\n * \n * \n * Groups elements from the
[Grouping] source by key and applies [operation] to the elements of each group sequentially,\n * passing the
previously accumulated value and the current element as arguments,\n * and stores the results in the given
[destination] map.\n * \n * The key for each element is provided by the [Grouping.keyOf] function.\n * \n *
@param operation a function that is invoked on each element with the following parameters:\n *     - `key`: the key of the group
this element belongs to;\n *     - `accumulator`: the current value of the accumulator of the group, can be `null` if it's
the first `element` encountered in the group;\n *     - `element`: the element from the source being aggregated;\n *     -
`first`: indicates whether it's the first `element` encountered in the group.\n * \n * If the [destination] map already has
a value corresponding to some key,\n * then the elements being aggregated for that key are never considered as
`first`.\n * \n * @return the [destination] map associating the key of each group with the result of aggregation of the
group elements.\n * \n * @sample samples.collections.Grouping.aggregateByRadixTo\n * \n * @SinceKotlin("1.1")\n * public inline fun <T, K, R, M : MutableMap<in K, R>> Grouping<T,
K>.aggregateTo(\n *     destination: M,\n *     operation: (key: K, accumulator: R?, element: T, first: Boolean) -> R\n * ): M
{\n *     for (e in this.sourceIterator()) {\n *         val key = keyOf(e)\n *         val accumulator = destination[key]\n *
destination[key] = operation(key, accumulator, e, accumulator == null && !destination.containsKey(key))\n *     }\n *
return destination\n * }\n * \n * \n * Groups elements from the [Grouping] source by key and applies [operation] to the
elements of each group sequentially,\n * passing the previously accumulated value and the current element as
arguments, and stores the results in a new map.\n * \n * An initial value of accumulator is provided by
[initialValueSelector] function.\n * \n * @param initialValueSelector a function that provides an initial value of
accumulator for each group.\n * \n * It's invoked with parameters:\n *     - `key`: the key of the group;\n *     - `element`: the
first element being encountered in that group.\n * \n * @param operation a function that is invoked on each element
with the following parameters:\n *     - `key`: the key of the group this element belongs to;\n *     - `accumulator`: the
current value of the accumulator of the group;\n *     - `element`: the element from the source being accumulated.\n *
*\n * @return a [Map] associating the key of each group with the result of accumulating the group elements.\n * \n *
@param sample samples.collections.Grouping.foldByEvenLengthWithComputedInitialValue\n * \n * @SinceKotlin("1.1")\n * public inline fun <T, K, R> Grouping<T, K>.fold(\n *     initialValueSelector: (key: K,
element: T) -> R,\n *     operation: (key: K, accumulator: R, element: T) -> R\n * ): Map<K, R> =\n * @Suppress("UNCHECKED_CAST")\n * aggregate { key, acc, e, first -> operation(key, if (first)
initialValueSelector(key, e) else acc as R, e) }\n * \n * \n * Groups elements from the [Grouping] source by key and
applies [operation] to the elements of each group sequentially,\n * passing the previously accumulated value and the
current element as arguments,\n * and stores the results in the given [destination] map.\n * \n * An initial value of
accumulator is provided by [initialValueSelector] function.\n * \n * @param initialValueSelector a function that
provides an initial value of accumulator for each group.\n * \n * It's invoked with parameters:\n *     - `key`: the key of the

```

group;\n \* - `element`: the first element being encountered in that group.\n \*\n \* If the [destination] map already has a value corresponding to some key, that value is used as an initial value of\n \*\n \* the accumulator for that group and the [initialValueSelector] function is not called for that group.\n \*\n \* @param operation a function that is invoked on each element with the following parameters:\n \*\n \* - `key`: the key of the group this element belongs to;\n \*\n \* - `accumulator`: the current value of the accumulator of the group;\n \*\n \* - `element`: the element from the source being accumulated.\n \*\n \* @return the [destination] map associating the key of each group with the result of accumulating the group elements.\n \*\n \* @sample

```

samples.collections.Grouping.foldByEvenLengthWithComputedInitialValueTo\n */\n@SinceKotlin("1.1")\npublic inline fun <T, K, R, M : MutableMap<in K, R>> Grouping<T, K>.foldTo(\n destination: M,\n initialValueSelector: (key: K, element: T) -> R,\n operation: (key: K, accumulator: R, element: T) -> R)\n): M =\n @Suppress("UNCHECKED_CAST")\n aggregateTo(destination) { key, acc, e, first -> operation(key, if (first) initialValueSelector(key, e) else acc as R, e) }\n\n\n/**\n * Groups elements from the [Grouping] source by key and applies [operation] to the elements of each group sequentially,\n * passing the previously accumulated value and the current element as arguments, and stores the results in a new map.\n * An initial value of accumulator is the same [initialValue] for each group.\n *\n * @param operation a function that is invoked on each element with the following parameters:\n * - `accumulator`: the current value of the accumulator of the group;\n * - `element`: the element from the source being accumulated.\n *\n * @return a [Map] associating the key of each group with the result of accumulating the group elements.\n *\n * @sample

```

```

samples.collections.Grouping.foldByEvenLengthWithConstantInitialValue\n */\n@SinceKotlin("1.1")\npublic inline fun <T, K, R> Grouping<T, K>.fold(\n initialValue: R,\n operation: (accumulator: R, element: T) -> R)\n): Map<K, R> =\n @Suppress("UNCHECKED_CAST")\n aggregate { _, acc, e, first -> operation(if (first) initialValue else acc as R, e) }\n\n\n/**\n * Groups elements from the [Grouping] source by key and applies [operation] to the elements of each group sequentially,\n * passing the previously accumulated value and the current element as arguments,\n * and stores the results in the given [destination] map.\n * An initial value of accumulator is the same [initialValue] for each group.\n *\n * If the [destination] map already has a value corresponding to the key of some group,\n * that value is used as an initial value of the accumulator for that group.\n *\n * @param operation a function that is invoked on each element with the following parameters:\n * - `accumulator`: the current value of the accumulator of the group;\n * - `element`: the element from the source being accumulated.\n *\n * @return the [destination] map associating the key of each group with the result of accumulating the group elements.\n *\n * @sample

```

```

samples.collections.Grouping.foldByEvenLengthWithConstantInitialValueTo\n */\n@SinceKotlin("1.1")\npublic inline fun <T, K, R, M : MutableMap<in K, R>> Grouping<T, K>.foldTo(\n destination: M,\n initialValue: R,\n operation: (accumulator: R, element: T) -> R)\n): M =\n @Suppress("UNCHECKED_CAST")\n aggregateTo(destination) { _, acc, e, first -> operation(if (first) initialValue else acc as R, e) }\n\n\n/**\n * Groups elements from the [Grouping] source by key and applies the reducing [operation] to the elements of each group\n * sequentially starting from the second element of the group,\n * passing the previously accumulated value and the current element as arguments,\n * and stores the results in a new map.\n * An initial value of accumulator is the first element of the group.\n *\n * @param operation a function that is invoked on each subsequent element of the group with the following parameters:\n * - `key`: the key of the group this element belongs to;\n * - `accumulator`: the current value of the accumulator of the group;\n * - `element`: the element from the source being accumulated.\n *\n * @return a [Map] associating the key of each group with the result of accumulating the group elements.\n *\n * @sample

```

```

samples.collections.Grouping.reduceByMaxVowels\n */\n@SinceKotlin("1.1")\npublic inline fun <S, T : S, K> Grouping<T, K>.reduce(\n operation: (key: K, accumulator: S, element: T) -> S)\n): Map<K, S> =\n aggregate { key, acc, e, first ->\n @Suppress("UNCHECKED_CAST")\n if (first) e else operation(key, acc as S, e)\n }\n\n\n/**\n * Groups elements from the [Grouping] source by key and applies the reducing [operation] to the elements of each group\n * sequentially starting from the second element of the group,\n * passing the previously accumulated value and the current element as arguments,\n * and stores the results in the given [destination] map.\n * An initial value of accumulator is the first element of the group.\n *\n * If the [destination] map already has a value corresponding to

```

the key of some group, `\n * that value is used as an initial value of the accumulator for that group and the first element of that group is also\n * subjected to the [operation].\n\n * @param operation a function that is invoked on each subsequent element of the group with the following parameters:\n * - `accumulator`: the current value of the accumulator of the group;\n * - `element`: the element from the source being folded;\n * \n * @return the [destination] map associating the key of each group with the result of accumulating the group elements.\n * @sample samples.collections.Grouping.reduceByMaxVowelsTo\n * ^\n * @SinceKotlin("1.1")\n public inline fun <S, T : S, K, M : MutableMap<in K, S>> Grouping<T, K>.reduceTo(destination: M, operation: (key: K, accumulator: S, element: T) -> S): M =\n aggregateTo(destination) { key, acc, e, first ->\n @Suppress("UNCHECKED_CAST")\n if (first) e else operation(key, acc as S, e)\n }\n\n\n * Groups elements from the [Grouping] source by key and counts elements in each group to the given [destination] map.\n * \n * If the [destination] map already has a value corresponding to the key of some group, \n * that value is used as an initial value of the counter for that group.\n * \n * @return the [destination] map associating the key of each group with the count of elements in the group.\n * \n * @sample samples.collections.Grouping.groupingByEachCount\n * ^\n * @SinceKotlin("1.1")\n public fun <T, K, M : MutableMap<in K, Int>> Grouping<T, K>.eachCountTo(destination: M): M =\n foldTo(destination, 0) { acc, _ -> acc + 1 }\n\n\n * Groups elements from the [Grouping] source by key and sums values provided by the [valueSelector] function for elements in each group \n * to the given [destination] map.\n * \n * \n * If the [destination] map already has a value corresponding to the key of some group, \n * that value is used as an initial value of the sum for that group.\n * \n * @return the [destination] map associating the key of each group with the sum of elements in the group.\n * \n * @sample samples.collections.Grouping.groupingByEachSum\n * ^\n * @SinceKotlin("1.1")\n public inline fun <T, K, M : MutableMap<in K, Int>> Grouping<T, K>.eachSumOfTo(destination: M, valueSelector: (T) -> Int): M =\n foldTo(destination, 0) { acc, e -> acc + valueSelector(e) }\n\n\n * // TODO: sum by long and by double overloads\n public inline fun <T, K, M : MutableMap<in K, Long>> Grouping<T, K>.sumEachByLongTo(destination: M, valueSelector: (T) -> Long): M =\n foldTo(destination, 0L) { acc, e -> acc + valueSelector(e) }\n\n\n public inline fun <T, K> Grouping<T, K>.sumEachByLong(valueSelector: (T) -> Long): Map<K, Long> =\n fold(0L) { acc, e -> acc + valueSelector(e) }\n\n\n public inline fun <T, K, M : MutableMap<in K, Double>> Grouping<T, K>.sumEachByDoubleTo(destination: M, valueSelector: (T) -> Double): M =\n foldTo(destination, 0.0) { acc, e -> acc + valueSelector(e) }\n\n\n public inline fun <T, K> Grouping<T, K>.sumEachByDouble(valueSelector: (T) -> Double): Map<K, Double> =\n fold(0.0) { acc, e -> acc + valueSelector(e) }\n\n\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors. Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file. \n * \n * An interface for indexing access to a collection of key-value pairs, where type of key is [String] and type of value is [Any?][Any]. \n * \n * @sample samples.collections.Json\n * \n * @SinceKotlin("1.1")\n public external interface Json {\n /** \n * Calls to the function will be translated to indexing operation (square brackets) on the receiver with [propertyName] as the argument. \n * \n * E.g. for next code:\n * \n * ````kotlin\n * fun test(j: Json, p: String) = j["prop"] + j.get(p)\n * \n * ````\n * \n * will be generated:\n * \n * ````js\n * function test(j, p) {\n * return j["prop"] + j[p];\n * }\n * \n * ````\n * \n * ^\n * \n * operator fun get(propertyName: String): Any? \n * \n * Calls of the function will be translated to an assignment of [value] to the receiver indexed (with square brackets/index operation) with [propertyName]. \n * \n * E.g. for the following code:\n * \n * ````kotlin\n * fun test(j: Json, p: String, newValue: Any) {\n * j["prop"] = 1\n * j.set(p, newValue)\n * }\n * \n * ````\n * \n * will be generated:\n * \n * ````js\n * function test(j, p, newValue) {\n * j["prop"] = 1;\n * j[p] = newValue;\n * }\n * \n * ````\n * \n * ^\n * \n * operator fun set(propertyName: String, value: Any?): Unit \n * \n * Returns a simple JavaScript object (as [Json]) using provided key-value pairs as names and values of its properties. \n * \n * @sample samples.collections.Json\n * \n * @SinceKotlin("1.1")\n public fun json(vararg pairs: Pair<String, Any?>): Json {\n val res: dynamic = js("{}")\n for ((name, value) in pairs) {\n res[name] = value\n }\n return res\n }\n\n\n * Adds key-value pairs from [other] to [this]. \n * \n * Returns the original receiver. \n * \n * @sample samples.collections.Json\n * \n * @SinceKotlin("1.1")\n public fun Json.add(other: Json): Json {\n val keys: Array<String> = js("Object").keys(other)\n for (key in keys) {\n if (other.asDynamic().hasOwnProperty(key)) {\n this[key] = other[key];\n }\n }\n return this\n }\n\n\n * Exposes the JavaScript [JSON object](https://developer.mozilla.org/en-`

US/docs/Web/JavaScript/Reference/Global\_Objects/JSON) to Kotlin.\n

```
*\n@Suppress("NOT_DOCUMENTED")\npublic external object JSON {\n    public fun stringify(o: Any?):\nString\n    public fun stringify(o: Any?, replacer: ((key: String, value: Any?) -> Any?): String\n    public fun\nstringify(o: Any?, replacer: ((key: String, value: Any?) -> Any?)? = definedExternally, space: Int): String\n    public\nfun stringify(o: Any?, replacer: ((key: String, value: Any?) -> Any?)? = definedExternally, space: String): String\n    public\nfun stringify(o: Any?, replacer: Array<String>): String\n    public fun stringify(o: Any?, replacer:\nArray<String>, space: Int): String\n    public fun stringify(o: Any?, replacer: Array<String>, space: String):\nString\n\n    public fun <T> parse(text: String): T\n    public fun <T> parse(text: String, reviver: ((key: String, value:\nAny?) -> Any?): T)\n}\n"/\n\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language\ncontributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the\nlicense/LICENSE.txt file.\n */\npackage kotlin.math\n\nimport kotlin.internal.InlineOnly\nimport kotlin.js.JsMath\nas nativeMath\n\n// region ===== Double Math
```

```
=====\n\n/** Computes the sine of the angle [x] given in\nradians.\n * Special cases:\n * - `sin(NaN|+Inf|-Inf)` is `NaN`\n
```

```
*\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun sin(x: Double): Double =
```

```
nativeMath.sin(x)\n\n/** Computes the cosine of the angle [x] given in radians.\n * Special cases:\n * -
```

```
`cos(NaN|+Inf|-Inf)` is `NaN`\n */\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun cos(x: Double):
```

```
Double = nativeMath.cos(x)\n\n/** Computes the tangent of the angle [x] given in radians.\n * Special cases:\n * -
```

```
`tan(NaN|+Inf|-Inf)` is `NaN`\n */\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun tan(x:
```

```
Double): Double = nativeMath.tan(x)\n\n/** Computes the arc sine of the value [x];\n * the returned value is an
```

```
angle in the range from  $-\pi/2$  to  $\pi/2$  radians.\n * Special cases:\n * - `asin(x)` is `NaN`, when  $abs(x) > 1$ 
```

```
or x is `NaN`\n */\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun asin(x: Double): Double =
```

```
nativeMath.asin(x)\n\n/** Computes the arc cosine of the value [x];\n * the returned value is an angle in the
```

```
range from  $0.0$  to  $\pi$  radians.\n * Special cases:\n * - `acos(x)` is `NaN`, when  $abs(x) > 1$  or x is `NaN`\n
```

```
*\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun acos(x: Double): Double =
```

```
nativeMath.acos(x)\n\n/** Computes the arc tangent of the value [x];\n * the returned value is an angle in the
```

```
range from  $-\pi/2$  to  $\pi/2$  radians.\n * Special cases:\n * - `atan(NaN)` is `NaN`\n
```

```
*\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun atan(x: Double): Double =
```

```
nativeMath.atan(x)\n\n/** Returns the angle  $\theta$  of the polar coordinates  $(r, \theta)$  that correspond
```

```
to the
```

```
rectangular coordinates  $(x, y)$  by computing the arc tangent of the value  $y / x$ ;\n * the returned value is an angle
```

```
in the range from  $-\pi$  to  $\pi$  radians.\n * Special cases:\n * - `atan2(0.0, 0.0)` is  $0.0$ \n * - `atan2(0.0, x)` is
```

```
 $0.0$  for  $x > 0$  and  $\pi$  for  $x < 0$ \n * - `atan2(-0.0, x)` is  $-0.0$  for  $x > 0$  and  $-\pi$  for  $x < 0$ \n * - `atan2(y,
```

```
+Inf)` is  $0.0$  for  $0 < y < +Inf$  and  $-0.0$  for  $-Inf < y < 0$ \n * - `atan2(y, -Inf)` is  $\pi$  for  $0 < y < +Inf$  and  $-\pi$ 
```

```
for  $-\pi < y < 0$ \n * - `atan2(y, 0.0)` is  $\pi/2$  for  $y > 0$  and  $-\pi/2$  for  $y < 0$ \n * - `atan2(+Inf, x)` is  $\pi/2$  for
```

```
finite  $x$ \n * - `atan2(-Inf, x)` is  $-\pi/2$  for finite  $x$ \n * - `atan2(NaN, x)` and `atan2(y, NaN)` is `NaN`\n
```

```
*\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun atan2(y: Double, x: Double): Double =
```

```
nativeMath.atan2(y, x)\n\n/** Computes the hyperbolic sine of the value [x].\n * Special cases:\n * -
```

```
`sinh(NaN)` is `NaN`\n * - `sinh(+Inf)` is  $+Inf$ \n * - `sinh(-Inf)` is  $-Inf$ \n
```

```
*\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun sinh(x: Double): Double =
```

```
nativeMath.sinh(x)\n\n/** Computes the hyperbolic cosine of the value [x].\n * Special cases:\n * -
```

```
`cosh(NaN)` is `NaN`\n * - `cosh(+Inf|-Inf)` is  $+Inf$ \n */\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual
```

```
inline fun cosh(x: Double): Double = nativeMath.cosh(x)\n\n/** Computes the hyperbolic tangent of the value
```

```
[x].\n * Special cases:\n * - `tanh(NaN)` is `NaN`\n * - `tanh(+Inf)` is  $1.0$ \n * - `tanh(-Inf)` is  $-1.0$ \n
```

```
*\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun tanh(x: Double): Double =
```

```
nativeMath.tanh(x)\n\n/** Computes the inverse hyperbolic sine of the value [x].\n * The returned value is
```

```
 $y$  such that  $\sinh(y) == x$ .\n * Special cases:\n * - `asinh(NaN)` is `NaN`\n * - `asinh(+Inf)` is  $+Inf$ \n * -
```

```
`asinh(-Inf)` is  $-Inf$ \n */\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun asinh(x: Double): Double =
```

```
nativeMath.asinh(x)\n\n/** Computes the inverse hyperbolic cosine of the value [x].\n * The returned
```

value is positive `y` such that  $\cosh(y) == x$ .  
Special cases:  
-  $\text{acosh}(\text{NaN})$  is  $\text{NaN}$   
-  $\text{acosh}(x)$  is  $\text{NaN}$  when  $x < 1$   
-  $\text{acosh}(+\text{Inf})$  is  $+\text{Inf}$   
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun acosh(x: Double): Double = nativeMath.acosh(x)  
Computes the inverse hyperbolic tangent of the value [x].  
The returned value is `y` such that  $\tanh(y) == x$ .  
Special cases:  
-  $\text{tanh}(\text{NaN})$  is  $\text{NaN}$   
-  $\text{tanh}(x)$  is  $\text{NaN}$  when  $x > 1$  or  $x < -1$   
-  $\text{tanh}(1.0)$  is  $+\text{Inf}$   
-  $\text{tanh}(-1.0)$  is  $-\text{Inf}$   
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun atanh(x: Double): Double = nativeMath.atanh(x)  
Computes  $\sqrt{x^2 + y^2}$  without intermediate overflow or underflow.  
Special cases:  
- returns  $+\text{Inf}$  if any of arguments is infinite  
- returns  $\text{NaN}$  if any of arguments is  $\text{NaN}$  and the other is not infinite  
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun hypot(x: Double, y: Double): Double = nativeMath.hypot(x, y)  
Computes the positive square root of the value [x].  
Special cases:  
-  $\sqrt{x}$  is  $\text{NaN}$  when  $x < 0$  or  $x$  is  $\text{NaN}$   
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun sqrt(x: Double): Double = nativeMath.sqrt(x)  
Computes Euler's number `e` raised to the power of the value [x].  
Special cases:  
-  $\text{exp}(\text{NaN})$  is  $\text{NaN}$   
-  $\text{exp}(+\text{Inf})$  is  $+\text{Inf}$   
-  $\text{exp}(-\text{Inf})$  is  $0.0$   
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun exp(x: Double): Double = nativeMath.exp(x)  
Computes  $\exp(x) - 1$ .  
This function can be implemented to produce more precise result for [x] near zero.  
Special cases:  
-  $\text{expm1}(\text{NaN})$  is  $\text{NaN}$   
-  $\text{expm1}(+\text{Inf})$  is  $+\text{Inf}$   
-  $\text{expm1}(-\text{Inf})$  is  $-1.0$   
@see [exp] function.  
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun expm1(x: Double): Double = nativeMath.expm1(x)  
Computes the logarithm of the value [x] to the given [base].  
Special cases:  
-  $\log(x, b)$  is  $\text{NaN}$  if either `x` or `b` are  $\text{NaN}$   
-  $\log(x, b)$  is  $\text{NaN}$  when  $x < 0$  or  $b \leq 0$  or  $b == 1.0$   
-  $\log(+\text{Inf}, +\text{Inf})$  is  $\text{NaN}$   
-  $\log(+\text{Inf}, b)$  is  $+\text{Inf}$  for  $b > 1$  and  $-\text{Inf}$  for  $b < 1$   
-  $\log(0.0, b)$  is  $-\text{Inf}$  for  $b > 1$  and  $+\text{Inf}$  for  $b > 1$   
See also logarithm functions for common fixed bases: [ln], [log10] and [log2].  
@SinceKotlin("1.2")@InlineOnly\npublic actual fun log(x: Double, base: Double): Double {  
if (base <= 0.0 || base == 1.0) return Double.NaN  
return nativeMath.log(x) / nativeMath.log(base)  
}
Computes the natural logarithm (base `E`) of the value [x].  
Special cases:  
-  $\ln(\text{NaN})$  is  $\text{NaN}$   
-  $\ln(x)$  is  $\text{NaN}$  when  $x < 0.0$   
-  $\ln(+\text{Inf})$  is  $+\text{Inf}$   
-  $\ln(0.0)$  is  $-\text{Inf}$   
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun ln(x: Double): Double = nativeMath.log(x)  
Computes the common logarithm (base 10) of the value [x].  
@see [ln] function for special cases.  
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun log10(x: Double): Double = nativeMath.log10(x)  
Computes the binary logarithm (base 2) of the value [x].  
@see [ln] function for special cases.  
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun log2(x: Double): Double = nativeMath.log2(x)  
Computes  $\ln(x + 1)$ .  
This function can be implemented to produce more precise result for [x] near zero.  
Special cases:  
-  $\ln1p(\text{NaN})$  is  $\text{NaN}$   
-  $\ln1p(x)$  is  $\text{NaN}$  where  $x < -1.0$   
-  $\ln1p(-1.0)$  is  $-\text{Inf}$   
-  $\ln1p(+\text{Inf})$  is  $+\text{Inf}$   
@see [ln] function  
@see [expm1] function  
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun ln1p(x: Double): Double = nativeMath.log1p(x)  
Rounds the given value [x] to an integer towards positive infinity.  
@return the smallest double value that is greater than or equal to the given value [x] and is a mathematical integer.  
Special cases:  
-  $\text{ceil}(x)$  is  $x$  where  $x$  is  $\text{NaN}$  or  $+\text{Inf}$  or  $-\text{Inf}$  or already a mathematical integer.  
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun ceil(x: Double): Double = nativeMath.ceil(x)  
Rounds the given value [x] to an integer towards negative infinity.  
@return the largest double value that is smaller than or equal to the given value [x] and is a mathematical integer.  
Special cases:  
-  $\text{floor}(x)$  is  $x$  where  $x$  is  $\text{NaN}$  or  $+\text{Inf}$  or  $-\text{Inf}$  or already a mathematical integer.  
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun floor(x: Double): Double = nativeMath.floor(x)  
Rounds the given value [x] to an integer towards zero.  
@return the value [x] having its fractional part truncated.  
Special cases:  
-  $\text{truncate}(x)$  is  $x$  where  $x$  is  $\text{NaN}$  or  $+\text{Inf}$  or  $-\text{Inf}$  or already a mathematical integer.  
@SinceKotlin("1.2")@InlineOnly\npublic actual inline fun truncate(x: Double): Double = nativeMath.trunc(x)  
Rounds the given value [x] towards the closest integer with ties rounded towards even integer.  
Special cases:  
-  $\text{round}(x)$  is  $x$  where  $x$  is  $\text{NaN}$  or  $+\text{Inf}$  or

```

`-Inf` or already a mathematical integer.\n *\n@SinceKotlin("1.2")\npublic actual fun round(x: Double): Double
{\n if (x % 0.5 != 0.0) {\n return nativeMath.round(x)\n }\n val floor = floor(x)\n return if (floor % 2 ==
0.0) floor else ceil(x)\n}\n\n/**\n * Returns the absolute value of the given value [x].\n *\n * Special cases:\n * -
`abs(NaN)` is `NaN`\n *\n * @see absoluteValue extension property for [Double]\n
*\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun abs(x: Double): Double =
nativeMath.abs(x)\n\n/**\n * Returns the sign of the given value [x]:\n * - `-1.0` if the value is negative,\n * - zero
if the value is zero,\n * - `1.0` if the value is positive\n *\n * Special case:\n * - `sign(NaN)` is `NaN`\n
*\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun sign(x: Double): Double =
nativeMath.sign(x)\n\n/**\n * Returns the smaller of two values.\n *\n * If either value is `NaN`, then the result is
`NaN`.\n *\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun min(a: Double, b: Double): Double =
nativeMath.min(a, b)\n\n/**\n * Returns the greater of two values.\n *\n * If either value is `NaN`, then the result is
`NaN`.\n *\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun max(a: Double, b: Double): Double =
nativeMath.max(a, b)\n\n// extensions\n\n/**\n * Raises this value to the power [x].\n *\n * Special cases:\n * -
`b.pow(0.0)` is `1.0`\n * - `b.pow(1.0) == b`\n * - `b.pow(NaN)` is `NaN`\n * - `NaN.pow(x)` is `NaN` for `x !=
0.0`\n * - `b.pow(Inf)` is `NaN` for `abs(b) == 1.0`\n * - `b.pow(x)` is `NaN` for `b < 0` and `x` is finite and not
an integer\n *\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun Double.pow(x: Double): Double =
nativeMath.pow(this, x)\n\n/**\n * Raises this value to the integer power [n].\n *\n * See the other overload of
[pow] for details.\n *\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun Double.pow(n: Int): Double
= nativeMath.pow(this, n.toDouble())\n\n/**\n * Returns the absolute value of this value.\n *\n * Special cases:\n *
- `NaN.absoluteValue` is `NaN`\n *\n * @see abs function\n *\n@SinceKotlin("1.2")\n@InlineOnly\npublic
actual inline val Double.absoluteValue: Double get() = nativeMath.abs(this)\n\n/**\n * Returns the sign of this
value:\n * - `-1.0` if the value is negative,\n * - zero if the value is zero,\n * - `1.0` if the value is positive\n *\n *
Special case:\n * - `NaN.sign` is `NaN`\n *\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline val
Double.sign: Double get() = nativeMath.sign(this)\n\n/**\n * Returns this value with the sign bit same as of the
[sign] value.\n *\n@SinceKotlin("1.2")\n@InlineOnly\npublic actual inline fun Double.withSign(sign: Int):
Double = this.withSign(sign.toDouble())\n\n/**\n * Returns the ulp (unit in the last place) of this value.\n *\n * An
ulp is a positive distance between this value and the next nearest [Double] value larger in magnitude.\n *\n * Special
Cases:\n * - `NaN.ulp` is `NaN`\n * - `x.ulp` is `+Inf` when `x` is `+Inf` or `-Inf`\n * - `0.0.ulp` is
`Double.MIN_VALUE`\n *\n@SinceKotlin("1.2")\npublic actual val Double.ulp: Double get() = when {\n this
< 0 -> (-this).ulp\n this.isNaN() || this == Double.POSITIVE_INFINITY -> this\n this ==
Double.MAX_VALUE -> this - this.nextDown()\n else -> this.nextUp() - this\n}\n\n/**\n * Returns the [Double]
value nearest to this value in direction of positive infinity.\n *\n@SinceKotlin("1.2")\npublic actual fun
Double.nextUp(): Double = when {\n this.isNaN() || this == Double.POSITIVE_INFINITY -> this\n this == 0.0
-> Double.MIN_VALUE\n else -> Double.fromBits(this.toRawBits() + if (this > 0) 1 else -1)\n}\n\n/**\n *
Returns the [Double] value nearest to this value in direction of negative infinity.\n
*\n@SinceKotlin("1.2")\npublic actual fun Double.nextDown(): Double = when {\n this.isNaN() || this ==
Double.NEGATIVE_INFINITY -> this\n this == 0.0 -> -Double.MIN_VALUE\n else ->
Double.fromBits(this.toRawBits() + if (this > 0) -1 else 1)\n}\n\n/**\n * Returns the [Double] value nearest to this
value in direction from this value towards the value [to].\n *\n * Special cases:\n * - `x.nextTowards(y)` is `NaN` if
either `x` or `y` are `NaN`\n * - `x.nextTowards(x) == x`\n *\n@SinceKotlin("1.2")\npublic actual fun
Double.nextTowards(to: Double): Double = when {\n this.isNaN() || to.isNaN() -> Double.NaN\n to == this ->
to\n to > this -> this.nextUp()\n else /* to < this */ -> this.nextDown()\n}\n\n/**\n * Rounds this [Double]
value to the nearest integer and converts the result to [Int].\n *\n * Ties are rounded towards positive infinity.\n *\n *
Special cases:\n * - `x.roundToInt() == Int.MAX_VALUE` when `x > Int.MAX_VALUE`\n * - `x.roundToInt()
== Int.MIN_VALUE` when `x < Int.MIN_VALUE`\n *\n * @throws IllegalArgumentException when this value is
`NaN`\n *\n@SinceKotlin("1.2")\npublic actual fun Double.roundToInt(): Int = when {\n isNaN() -> throw
IllegalArgumentException("Cannot round NaN value.")\n this > Int.MAX_VALUE -> Int.MAX_VALUE\n this < Int.MIN_VALUE
-> Int.MIN_VALUE\n else -> nativeMath.round(this).toInt()\n}\n\n/**\n * Rounds this

```

[Double] value to the nearest integer and converts the result to [Long].  
 \* Ties are rounded towards positive infinity.  
 \* Special cases:  
 \*  $x.\text{roundToLong}() == \text{Long.MAX\_VALUE}$  when  $x > \text{Long.MAX\_VALUE}$   
 \*  $x.\text{roundToLong}() == \text{Long.MIN\_VALUE}$  when  $x < \text{Long.MIN\_VALUE}$   
 \* @throws IllegalArgumentException when this value is NaN  
 \* @SinceKotlin("1.2")  
 \* public actual fun Double.roundToLong(): Long = when {  
 \* isNaN() -> throw IllegalArgumentException("Cannot round NaN value.")  
 \* this > Long.MAX\_VALUE -> Long.MAX\_VALUE  
 \* this < Long.MIN\_VALUE -> Long.MIN\_VALUE  
 \* else -> nativeMath.round(this).toLong()  
 \* }  
 \* // endregion  
 \* // region  
 \* ===== Float Math =====  
 \* Computes the sine of the angle [x] given in radians.  
 \* \* Special cases:  
 \* \*  $\sin(\text{NaN}|\pm\text{Inf}|\text{Inf})$  is NaN  
 \* \* @SinceKotlin("1.2")  
 \* @InlineOnly  
 \* public actual inline fun sin(x: Float): Float = nativeMath.sin(x.toDouble()).toFloat()  
 \* Computes the cosine of the angle [x] given in radians.  
 \* \* Special cases:  
 \* \*  $\cos(\text{NaN}|\pm\text{Inf}|\text{Inf})$  is NaN  
 \* \* @SinceKotlin("1.2")  
 \* @InlineOnly  
 \* public actual inline fun cos(x: Float): Float = nativeMath.cos(x.toDouble()).toFloat()  
 \* Computes the tangent of the angle [x] given in radians.  
 \* \* Special cases:  
 \* \*  $\tan(\text{NaN}|\pm\text{Inf}|\text{Inf})$  is NaN  
 \* \* @SinceKotlin("1.2")  
 \* @InlineOnly  
 \* public actual inline fun tan(x: Float): Float = nativeMath.tan(x.toDouble()).toFloat()  
 \* Computes the arc sine of the value [x];  
 \* \* the returned value is an angle in the range from  $-\text{PI}/2$  to  $\text{PI}/2$  radians.  
 \* \* Special cases:  
 \* \*  $\text{asin}(x)$  is NaN, when  $|\text{abs}(x)| > 1$  or x is NaN  
 \* \* @SinceKotlin("1.2")  
 \* @InlineOnly  
 \* public actual inline fun asin(x: Float): Float = nativeMath.asin(x.toDouble()).toFloat()  
 \* Computes the arc cosine of the value [x];  
 \* \* the returned value is an angle in the range from  $0.0$  to  $\text{PI}$  radians.  
 \* \* Special cases:  
 \* \*  $\text{acos}(x)$  is NaN, when  $|\text{abs}(x)| > 1$  or x is NaN  
 \* \* @SinceKotlin("1.2")  
 \* @InlineOnly  
 \* public actual inline fun acos(x: Float): Float = nativeMath.acos(x.toDouble()).toFloat()  
 \* Computes the arc tangent of the value [x];  
 \* \* the returned value is an angle in the range from  $-\text{PI}/2$  to  $\text{PI}/2$  radians.  
 \* \* Special cases:  
 \* \*  $\text{atan}(\text{NaN})$  is NaN  
 \* \* @SinceKotlin("1.2")  
 \* @InlineOnly  
 \* public actual inline fun atan(x: Float): Float = nativeMath.atan(x.toDouble()).toFloat()  
 \* Returns the angle  $\theta$  of the polar coordinates (r, theta) that correspond  
 \* to the rectangular coordinates (x, y) by computing the arc tangent of the value  $y/x$ ;  
 \* \* the returned value is an angle in the range from  $-\text{PI}$  to  $\text{PI}$  radians.  
 \* \* Special cases:  
 \* \*  $\text{atan2}(0.0, 0.0)$  is  $0.0$   
 \* \*  $\text{atan2}(0.0, x)$  is  $0.0$  for  $x > 0$  and  $\text{PI}$  for  $x < 0$   
 \* \*  $\text{atan2}(-0.0, x)$  is  $-0.0$  for  $x > 0$  and  $-\text{PI}$  for  $x < 0$   
 \* \*  $\text{atan2}(y, +\text{Inf})$  is  $0.0$  for  $0 < y < +\text{Inf}$  and  $-0.0$  for  $-\text{Inf} < y < 0$   
 \* \*  $\text{atan2}(y, -\text{Inf})$  is  $\text{PI}$  for  $0 < y < +\text{Inf}$  and  $-\text{PI}$  for  $-\text{Inf} < y < 0$   
 \* \*  $\text{atan2}(y, 0.0)$  is  $\text{PI}/2$  for  $y > 0$  and  $-\text{PI}/2$  for  $y < 0$   
 \* \*  $\text{atan2}(+\text{Inf}, x)$  is  $\text{PI}/2$  for finite  $x$   
 \* \*  $\text{atan2}(-\text{Inf}, x)$  is  $-\text{PI}/2$  for finite  $x$   
 \* \*  $\text{atan2}(\text{NaN}, x)$  and  $\text{atan2}(y, \text{NaN})$  is NaN  
 \* \* @SinceKotlin("1.2")  
 \* @InlineOnly  
 \* public actual inline fun atan2(y: Float, x: Float): Float = nativeMath.atan2(y.toDouble(), x.toDouble()).toFloat()  
 \* Computes the hyperbolic sine of the value [x].  
 \* \* Special cases:  
 \* \*  $\sinh(\text{NaN})$  is NaN  
 \* \*  $\sinh(+\text{Inf})$  is  $+\text{Inf}$   
 \* \*  $\sinh(-\text{Inf})$  is  $-\text{Inf}$   
 \* \* @SinceKotlin("1.2")  
 \* @InlineOnly  
 \* public actual inline fun sinh(x: Float): Float = nativeMath.sinh(x.toDouble()).toFloat()  
 \* Computes the hyperbolic cosine of the value [x].  
 \* \* Special cases:  
 \* \*  $\cosh(\text{NaN})$  is NaN  
 \* \*  $\cosh(+\text{Inf}|\text{Inf})$  is  $+\text{Inf}$   
 \* \* @SinceKotlin("1.2")  
 \* @InlineOnly  
 \* public actual inline fun cosh(x: Float): Float = nativeMath.cosh(x.toDouble()).toFloat()  
 \* Computes the hyperbolic tangent of the value [x].  
 \* \* Special cases:  
 \* \*  $\tanh(\text{NaN})$  is NaN  
 \* \*  $\tanh(+\text{Inf})$  is  $1.0$   
 \* \*  $\tanh(-\text{Inf})$  is  $-1.0$   
 \* \* @SinceKotlin("1.2")  
 \* @InlineOnly  
 \* public actual inline fun tanh(x: Float): Float = nativeMath.tanh(x.toDouble()).toFloat()  
 \* Computes the inverse hyperbolic sine of the value [x].  
 \* \* The returned value is  $y$  such that  $\sinh(y) == x$ .  
 \* \* Special cases:  
 \* \*  $\text{asinh}(\text{NaN})$  is NaN  
 \* \*  $\text{asinh}(+\text{Inf})$  is  $+\text{Inf}$   
 \* \*  $\text{asinh}(-\text{Inf})$  is  $-\text{Inf}$   
 \* \* @SinceKotlin("1.2")  
 \* @InlineOnly  
 \* public actual inline fun asinh(x: Float): Float = nativeMath.asinh(x.toDouble()).toFloat()  
 \* Computes the inverse hyperbolic cosine of the value [x].  
 \* \* The returned value is positive  $y$  such that  $\cosh(y) == x$ .  
 \* \* Special cases:  
 \* \*  $\text{acosh}(\text{NaN})$  is NaN  
 \* \*  $\text{acosh}(x)$  is NaN when  $x < 1$   
 \* \*  $\text{acosh}(+\text{Inf})$  is  $+\text{Inf}$   
 \* \* @SinceKotlin("1.2")  
 \* @InlineOnly  
 \* public actual inline fun acosh(x: Float): Float =

`nativeMath.acosh(x.toDouble()).toFloat()` Computes the inverse hyperbolic tangent of the value [x].  
 \* The returned value is `y` such that  $\tanh(y) = x$ .  
 \* Special cases:  
 -  $\tanh(\text{NaN})$  is  $\text{NaN}$   
 -  $\tanh(x)$  is  $\text{NaN}$  when  $x > 1$  or  $x < -1$   
 -  $\tanh(1.0)$  is  $+\text{Inf}$   
 -  $\tanh(-1.0)$  is  $-\text{Inf}$

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun atanh(x: Float): Float =`  
`nativeMath.atanh(x.toDouble()).toFloat()` Computes  $\sqrt{x^2 + y^2}$  without intermediate overflow or underflow.  
 \* Special cases:  
 - returns  $+\text{Inf}$  if any of arguments is infinite  
 - returns  $\text{NaN}$  if any of arguments is  $\text{NaN}$  and the other is not infinite

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun hypot(x: Float, y: Float): Float =`  
`nativeMath.hypot(x.toDouble(), y.toDouble()).toFloat()` Computes the positive square root of the value [x].  
 \* Special cases:  
 -  $\sqrt{x}$  is  $\text{NaN}$  when  $x < 0$  or  $x$  is  $\text{NaN}$

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun sqrt(x: Float): Float =`  
`nativeMath.sqrt(x.toDouble()).toFloat()` Computes Euler's number `e` raised to the power of the value [x].  
 \* Special cases:  
 -  $\exp(\text{NaN})$  is  $\text{NaN}$   
 -  $\exp(+\text{Inf})$  is  $+\text{Inf}$   
 -  $\exp(-\text{Inf})$  is  $0.0$

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun exp(x: Float): Float =`  
`nativeMath.exp(x.toDouble()).toFloat()` Computes  $\exp(x) - 1$ .  
 \* This function can be implemented to produce more precise result for [x] near zero.  
 \* Special cases:  
 -  $\expm1(\text{NaN})$  is  $\text{NaN}$   
 -  $\expm1(+\text{Inf})$  is  $+\text{Inf}$   
 -  $\expm1(-\text{Inf})$  is  $-1.0$   
 \* @see [exp] function.

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun expm1(x: Float): Float =`  
`nativeMath.expm1(x.toDouble()).toFloat()` Computes the logarithm of the value [x] to the given [base].  
 \* Special cases:  
 -  $\log(x, b)$  is  $\text{NaN}$  if either `x` or `b` are  $\text{NaN}$   
 -  $\log(x, b)$  is  $\text{NaN}$  when  $x < 0$  or  $b \leq 0$  or  $b = 1.0$   
 -  $\log(+\text{Inf}, +\text{Inf})$  is  $\text{NaN}$   
 -  $\log(+\text{Inf}, b)$  is  $+\text{Inf}$  for  $b > 1$  and  $-\text{Inf}$  for  $b < 1$   
 -  $\log(0.0, b)$  is  $-\text{Inf}$  for  $b > 1$  and  $+\text{Inf}$  for  $b > 1$   
 \* See also logarithm functions for common fixed bases: [ln], [log10] and [log2].

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun log(x: Float, base: Float): Float =`  
`log(x.toDouble(), base.toDouble()).toFloat()` Computes the natural logarithm (base `E`) of the value [x].  
 \* Special cases:  
 -  $\ln(\text{NaN})$  is  $\text{NaN}$   
 -  $\ln(x)$  is  $\text{NaN}$  when  $x < 0.0$   
 -  $\ln(+\text{Inf})$  is  $+\text{Inf}$   
 -  $\ln(0.0)$  is  $-\text{Inf}$

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun ln(x: Float): Float =`  
`nativeMath.log(x.toDouble()).toFloat()` Computes the common logarithm (base 10) of the value [x].  
 \* @see [ln] function for special cases.

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun log10(x: Float): Float =`  
`nativeMath.log10(x.toDouble()).toFloat()` Computes the binary logarithm (base 2) of the value [x].  
 \* @see [ln] function for special cases.

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun log2(x: Float): Float =`  
`nativeMath.log2(x.toDouble()).toFloat()` Computes  $\ln(a + 1)$ .  
 \* This function can be implemented to produce more precise result for [x] near zero.  
 \* Special cases:  
 -  $\ln1p(\text{NaN})$  is  $\text{NaN}$   
 -  $\ln1p(x)$  is  $\text{NaN}$  where  $x < -1.0$   
 -  $\ln1p(-1.0)$  is  $-\text{Inf}$   
 -  $\ln1p(+\text{Inf})$  is  $+\text{Inf}$   
 \* @see [ln] function  
 \* @see [expm1] function

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun ln1p(x: Float): Float =`  
`nativeMath.log1p(x.toDouble()).toFloat()` Rounds the given value [x] to an integer towards positive infinity.  
 \* @return the smallest Float value that is greater than or equal to the given value [x] and is a mathematical integer.  
 \* Special cases:  
 -  $\text{ceil}(x)$  is  $x$  where  $x$  is  $\text{NaN}$  or  $+\text{Inf}$  or  $-\text{Inf}$  or already a mathematical integer.

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun ceil(x: Float): Float =`  
`nativeMath.ceil(x.toDouble()).toFloat()` Rounds the given value [x] to an integer towards negative infinity.  
 \* @return the largest Float value that is smaller than or equal to the given value [x] and is a mathematical integer.  
 \* Special cases:  
 -  $\text{floor}(x)$  is  $x$  where  $x$  is  $\text{NaN}$  or  $+\text{Inf}$  or  $-\text{Inf}$  or already a mathematical integer.

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun floor(x: Float): Float =`  
`nativeMath.floor(x.toDouble()).toFloat()` Rounds the given value [x] to an integer towards zero.  
 \* @return the value [x] having its fractional part truncated.  
 \* Special cases:  
 -  $\text{truncate}(x)$  is  $x$  where  $x$  is  $\text{NaN}$  or  $+\text{Inf}$  or  $-\text{Inf}$  or already a mathematical integer.

`@SinceKotlin("1.2")@InlineOnly@public actual inline fun truncate(x: Float): Float =`  
`truncate(x.toDouble()).toFloat()` Rounds the given value [x] towards the closest integer with ties rounded towards even integer.  
 \* Special cases:  
 -  $\text{round}(x)$  is  $x$  where  $x$  is  $\text{NaN}$  or  $+\text{Inf}$  or  $-\text{Inf}$  or already a mathematical integer.

```

*^@SinceKotlin("1.2")^@InlineOnly^public actual inline fun round(x: Float): Float =
round(x.toDouble()).toFloat()\n\n/**\n * Returns the absolute value of the given value [x].\n * Special cases:\n * - `abs(NaN)` is `NaN`\n * @see absoluteValue extension property for [Float]\n
*^@SinceKotlin("1.2")^@InlineOnly^public actual inline fun abs(x: Float): Float =
nativeMath.abs(x.toDouble()).toFloat()\n\n/**\n * Returns the sign of the given value [x]:\n * - `1.0` if the value is
negative,\n * - zero if the value is zero,\n * - `1.0` if the value is positive\n * Special case:\n * - `sign(NaN)`
is `NaN`\n *^@SinceKotlin("1.2")^@InlineOnly^public actual inline fun sign(x: Float): Float =
nativeMath.sign(x.toDouble()).toFloat()\n\n\n/**\n * Returns the smaller of two values.\n * If either value is
`NaN`, then the result is `NaN`.\n *^@SinceKotlin("1.2")^@InlineOnly^public actual inline fun min(a: Float, b:
Float): Float = nativeMath.min(a, b)\n\n\n/**\n * Returns the greater of two values.\n * If either value is `NaN`,
then the result is `NaN`.\n *^@SinceKotlin("1.2")^@InlineOnly^public actual inline fun max(a: Float, b: Float):
Float = nativeMath.max(a, b)\n\n// extensions\n\n\n/**\n * Raises this value to the power [x].\n * Special
cases:\n * - `b.pow(0.0)` is `1.0`\n * - `b.pow(1.0) == b`\n * - `b.pow(NaN)` is `NaN`\n * - `NaN.pow(x)` is
`NaN` for `x != 0.0`\n * - `b.pow(Inf)` is `NaN` for `abs(b) == 1.0`\n * - `b.pow(x)` is `NaN` for `b < 0` and `x` is
finite and not an integer\n *^@SinceKotlin("1.2")^@InlineOnly^public actual inline fun Float.pow(x: Float):
Float = nativeMath.pow(this.toDouble(), x.toDouble()).toFloat()\n\n\n/**\n * Raises this value to the integer power
[n].\n * See the other overload of [pow] for details.\n *^@SinceKotlin("1.2")^@InlineOnly^public actual
inline fun Float.pow(n: Int): Float = nativeMath.pow(this.toDouble(), n.toDouble()).toFloat()\n\n\n/**\n * Returns the
absolute value of this value.\n * Special cases:\n * - `NaN.absoluteValue` is `NaN`\n * @see abs function\n
*^@SinceKotlin("1.2")^@InlineOnly^public actual inline val Float.absoluteValue: Float get() =
nativeMath.abs(this.toDouble()).toFloat()\n\n\n/**\n * Returns the sign of this value:\n * - `1.0` if the value is
negative,\n * - zero if the value is zero,\n * - `1.0` if the value is positive\n * Special case:\n * - `NaN.sign` is
`NaN`\n *^@SinceKotlin("1.2")^@InlineOnly^public actual inline val Float.sign: Float get() =
nativeMath.sign(this.toDouble()).toFloat()\n\n\n/**\n * Returns this value with the sign bit same as of the [sign]
value.\n * If [sign] is `NaN` the sign of the result is undefined.\n
*^@SinceKotlin("1.2")^@InlineOnly^public actual inline fun Float.withSign(sign: Float): Float =
this.toDouble().withSign(sign.toDouble()).toFloat()\n\n\n/**\n * Returns this value with the sign bit same as of the
[sign] value.\n *^@SinceKotlin("1.2")^@InlineOnly^public actual inline fun Float.withSign(sign: Int): Float =
this.toDouble().withSign(sign.toDouble()).toFloat()\n\n\n/**\n * Rounds this [Float] value to the nearest integer and
converts the result to [Int].\n * Ties are rounded towards positive infinity.\n * Special cases:\n * -
`x.roundToInt() == Int.MAX_VALUE` when `x > Int.MAX_VALUE`\n * - `x.roundToInt() == Int.MIN_VALUE`
when `x < Int.MIN_VALUE`\n * @throws IllegalArgumentException when this value is `NaN`\n
*^@SinceKotlin("1.2")^@InlineOnly^public actual inline fun Float.roundToInt(): Int =
toDouble().roundToInt()\n\n\n/**\n * Rounds this [Float] value to the nearest integer and converts the result to
[Long].\n * Ties are rounded towards positive infinity.\n * Special cases:\n * - `x.roundToLong() ==
Long.MAX_VALUE` when `x > Long.MAX_VALUE`\n * - `x.roundToLong() == Long.MIN_VALUE` when `x
< Long.MIN_VALUE`\n * @throws IllegalArgumentException when this value is `NaN`\n
*^@SinceKotlin("1.2")^@InlineOnly^public actual inline fun Float.roundToLong(): Long =
toDouble().roundToLong()\n\n\n// endregion\n\n// region ===== Integer Math
=====
\n\n\n/**\n * Returns the absolute value of the given value
[n].\n * Special cases:\n * - `abs(Int.MIN_VALUE)` is `Int.MIN_VALUE` due to an overflow\n * @see
absoluteValue extension property for [Int]\n *^@// TODO: remove manual 'or' when KT-19290 is
fixed\n
*^@SinceKotlin("1.2")^public actual fun abs(n: Int): Int = if (n < 0) (-n or 0) else n\n\n\n/**\n * Returns the
smaller of two values.\n *^@SinceKotlin("1.2")^@InlineOnly^public actual inline fun min(a: Int, b: Int): Int =
nativeMath.min(a, b)\n\n\n/**\n * Returns the greater of two values.\n
*^@SinceKotlin("1.2")^@InlineOnly^public actual inline fun max(a: Int, b: Int): Int = nativeMath.max(a,
b)\n\n\n/**\n * Returns the absolute value of this value.\n * Special cases:\n * -
`Int.MIN_VALUE.absoluteValue` is `Int.MIN_VALUE` due to an overflow\n * @see abs function\n

```

```

*^@SinceKotlin("1.2")^@InlineOnly^public actual inline val Int.absoluteValue: Int get() = abs(this)^n/n/**^n *
Returns the sign of this value:^n * -`-1` if the value is negative,^n * -`0` if the value is zero,^n * -`1` if the value
is positive^
*^@SinceKotlin("1.2")^public actual val Int.sign: Int get() = when {^n this < 0 -> -1^n this > 0 -
> 1^n else -> 0^n}^n/n/n/**^n * Returns the absolute value of the given value [n].^n *^n * Special cases:^n * -
`abs(Long.MIN_VALUE)` is `Long.MIN_VALUE` due to an overflow^
*^n * @see absoluteValue extension
property for [Long]^
*^@SinceKotlin("1.2")^public actual fun abs(n: Long): Long = if (n < 0) -n else
n^
n/n/**^n * Returns the smaller of two values.^
*^@SinceKotlin("1.2")^@Suppress("NOTHING_TO_INLINE")^public actual inline fun min(a: Long, b:
Long): Long = if (a <= b) a else b^
n/n/**^n * Returns the greater of two values.^
*^@SinceKotlin("1.2")^@Suppress("NOTHING_TO_INLINE")^public actual inline fun max(a: Long, b:
Long): Long = if (a >= b) a else b^
n/n/**^n * Returns the absolute value of this value.^
*^n * Special cases:^n * -
`Long.MIN_VALUE.absoluteValue` is `Long.MIN_VALUE` due to an overflow^
*^n * @see abs function^
*^@SinceKotlin("1.2")^@InlineOnly^public actual inline val Long.absoluteValue: Long get() =
abs(this)^
n/n/**^n * Returns the sign of this value:^n * -`-1` if the value is negative,^n * -`0` if the value is zero,^
n * -`1` if the value is positive^
*^@SinceKotlin("1.2")^public actual val Long.sign: Int get() = when {^n this
< 0 -> -1^n this > 0 -> 1^n else -> 0^n}^
n/n/n// endregion^n","/^
*^n * Copyright 2010-2021 JetBrains s.r.o. and
Kotlin Programming Language contributors.^
*^n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.^
*^n/npackage kotlin^
n/n/**^n * Returns `true` if the specified
number is a^
n * Not-a-Number (NaN) value, `false` otherwise.^
*^n/npublic actual fun Double.isNaN(): Boolean =
this != this^
n/n/**^n * Returns `true` if the specified number is a^
n * Not-a-Number (NaN) value, `false` otherwise.^
n/npublic actual fun Float.isNaN(): Boolean = this != this^
n/n/**^n * Returns `true` if this value is infinitely large in
magnitude.^
*^n/npublic actual fun Double.isInfinite(): Boolean = this == Double.POSITIVE_INFINITY || this ==
Double.NEGATIVE_INFINITY^
n/n/**^n * Returns `true` if this value is infinitely large in magnitude.^
*^n/npublic
actual fun Float.isInfinite(): Boolean = this == Float.POSITIVE_INFINITY || this ==
Float.NEGATIVE_INFINITY^
n/n/**^n * Returns `true` if the argument is a finite floating-point value; returns
`false` otherwise (for `NaN` and infinity arguments).^
*^n/npublic actual fun Double.isFinite(): Boolean =
!isInfinite() && !isNaN()^
n/n/**^n * Returns `true` if the argument is a finite floating-point value; returns `false`
otherwise (for `NaN` and infinity arguments).^
*^n/npublic actual fun Float.isFinite(): Boolean = !isInfinite() &&
!isNaN()^
n/n/n/**^n * Counts the number of set bits in the binary representation of this [Int] number.^
*^@SinceKotlin("1.4")^@WasExperimental(ExperimentalStdlibApi::class)^public actual fun
Int.countOneBits(): Int {^n // Hacker's Delight 5-1 algorithm^
n var v = this^
n v = (v and 0x55555555) +
(v.ushr(1) and 0x55555555)^
n v = (v and 0x33333333) + (v.ushr(2) and 0x33333333)^
n v = (v and 0x0F0F0F0F)^
+ (v.ushr(4) and 0x0F0F0F0F)^
n v = (v and 0x00FF00FF) + (v.ushr(8) and 0x00FF00FF)^
n v = (v and
0x0000FFFF) + (v.ushr(16))^
n return v^n}^
n/n/**^n * Counts the number of consecutive most significant bits that
are zero in the binary representation of this [Int] number.^
*^@SinceKotlin("1.4")^@WasExperimental(ExperimentalStdlibApi::class)^@kotlin.internal.InlineOnly^publi
c actual inline fun Int.countLeadingZeroBits(): Int = JsMath.clz32(this)^
n/n/**^n * Counts the number of
consecutive least significant bits that are zero in the binary representation of this [Int] number.^
*^@SinceKotlin("1.4")^@WasExperimental(ExperimentalStdlibApi::class)^public actual fun
Int.countTrailingZeroBits(): Int =^
n // Hacker's Delight 5-4 algorithm for expressing countTrailingZeroBits with
countLeadingZeroBits^
n Int.SIZE_BITS - (this or -this).inv().countLeadingZeroBits()^
n/n/**^n * Returns a
number having a single bit set in the position of the most significant set bit of this [Int] number,^
n * or zero, if this
number is zero.^
*^@SinceKotlin("1.4")^@WasExperimental(ExperimentalStdlibApi::class)^public actual fun
Int.takeHighestOneBit(): Int =^
n if (this == 0) 0 else 1.shl(Int.SIZE_BITS - 1 - countLeadingZeroBits())^
n/n/**^n *
Returns a number having a single bit set in the position of the least significant set bit of this [Int] number,^
n * or
zero, if this number is zero.^
*^@SinceKotlin("1.4")^@WasExperimental(ExperimentalStdlibApi::class)^public actual fun
Int.takeLowestOneBit(): Int =^
n // Hacker's Delight 2-1 algorithm for isolating rightmost 1-bit^
n this and -

```

this

`Int.rotateLeft(bitCount: Int): Int` = `shl(bitCount)` or `ushr(Int.SIZE_BITS - bitCount)`

Rotates the binary representation of this [Int] number left by the specified [bitCount] number of bits. The most significant bits pushed out from the left side reenter the number as the least significant bits on the right side. Rotating the number left by a negative bit count is the same as rotating it right by the negated bit count: `number.rotateLeft(-n) == number.rotateRight(n)`

Rotating by a multiple of [Int.SIZE\_BITS] (32) returns the same number, or more generally `number.rotateLeft(n) == number.rotateLeft(n % 32)`

`Int.rotateRight(bitCount: Int): Int` = `shl(Int.SIZE_BITS - bitCount)` or `ushr(bitCount)`

Rotates the binary representation of this [Int] number right by the specified [bitCount] number of bits. The least significant bits pushed out from the right side reenter the number as the most significant bits on the left side. Rotating the number right by a negative bit count is the same as rotating it left by the negated bit count: `number.rotateRight(-n) == number.rotateLeft(n)`

Rotating by a multiple of [Int.SIZE\_BITS] (32) returns the same number, or more generally `number.rotateRight(n) == number.rotateRight(n % 32)`

`Long.countOneBits(): Int` = `high.countOneBits() + low.countOneBits()`

Counts the number of consecutive most significant bits that are zero in the binary representation of this [Long] number.

`Long.countLeadingZeroBits(): Int` = `when (val high = this.high) { 0 -> Int.SIZE_BITS + low.countLeadingZeroBits() } else -> high.countLeadingZeroBits()`

Counts the number of consecutive least significant bits that are zero in the binary representation of this [Long] number.

`Long.countTrailingZeroBits(): Int` = `when (val low = this.low) { 0 -> Int.SIZE_BITS + high.countTrailingZeroBits() } else -> low.countTrailingZeroBits()`

Returns a number having a single bit set in the position of the most significant set bit of this [Long] number, or zero, if this number is zero.

`Long.takeHighestOneBit(): Long` = `when (val high = this.high) { 0 -> Long(low.takeHighestOneBit(), 0) } else -> Long(0, high.takeHighestOneBit())`

Returns a number having a single bit set in the position of the least significant set bit of this [Long] number, or zero, if this number is zero.

`Long.takeLowestOneBit(): Long` = `when (val low = this.low) { 0 -> Long(0, high.takeLowestOneBit()) } else -> Long(low.takeLowestOneBit(), 0)`

Rotates the binary representation of this [Long] number left by the specified [bitCount] number of bits. The most significant bits pushed out from the left side reenter the number as the least significant bits on the right side. Rotating the number left by a negative bit count is the same as rotating it right by the negated bit count: `number.rotateLeft(-n) == number.rotateRight(n)`

Rotating by a multiple of [Long.SIZE\_BITS] (64) returns the same number, or more generally `number.rotateLeft(n) == number.rotateLeft(n % 64)`

`Long.rotateLeft(bitCount: Int): Long` = `if ((bitCount and 31) != 0) { val low = this.low; val high = this.high; val newLow = low.shl(bitCount) or high.ushr(-bitCount); val newHigh = high.shl(bitCount) or low.ushr(-bitCount); return if ((bitCount and 32) == 0) Long(newLow, newHigh) else Long(newHigh, newLow) } else { return if ((bitCount and 32) == 0) this else Long(high, low) }`

Rotates the binary representation of this [Long] number right by the specified [bitCount] number of bits. The least significant bits pushed out from the right side reenter the number as the most significant bits on the left side. Rotating the number right by a negative bit count is the same as rotating it left by the negated bit count: `number.rotateRight(-n) == number.rotateLeft(n)`

Rotating by a multiple of [Long.SIZE\_BITS] (64) returns the same number, or more generally `number.rotateRight(n) == number.rotateRight(n % 64)`

```

*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic actual inline fun Long.rotateRight(bitCount: Int): Long = rotateLeft(-bitCount)\n", "/*\n * Copyright 2010-2018
JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.js\n\nimport
kotlin.internal.LowPriorityInOverloadResolution\n\n/**\n * Exposes the JavaScript [Promise
object](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Promise) to Kotlin.\n
*\n@Suppress("NOT_DOCUMENTED")\npublic open external class Promise<out T>(executor: (resolve: (T) ->
Unit, reject: (Throwable) -> Unit) -> Unit) {\n @LowPriorityInOverloadResolution\n public open fun <S>
then(onFulfilled: ((T) -> S)?): Promise<S>\n\n @LowPriorityInOverloadResolution\n public open fun <S>
then(onFulfilled: ((T) -> S)?, onRejected: ((Throwable) -> S)?): Promise<S>\n\n public open fun <S>
catch(onRejected: (Throwable) -> S): Promise<S>\n\n companion object {\n public fun <S> all(promise:
Array<out Promise<S>>): Promise<Array<out S>>\n\n public fun <S> race(promise: Array<out
Promise<S>>): Promise<S>\n\n public fun reject(e: Throwable): Promise<Nothing>\n\n public fun <S>
resolve(e: S): Promise<S>\n\n public fun <S> resolve(e: Promise<S>): Promise<S>\n } }\n\n// It's workaround
for KT-19672 since we can fix it properly until KT-11265 isn't fixed.\n\ninline fun <T, S>
Promise<Promise<T>>.then(\n noinline onFulfilled: ((T) -> S)?\n): Promise<S> {\n return
this.unsafeCast<Promise<T>>().then(onFulfilled)\n}\n\ninline fun <T, S> Promise<Promise<T>>.then(\n noinline
onFulfilled: ((T) -> S)?,\n noinline onRejected: ((Throwable) -> S)?\n): Promise<S> {\n return
this.unsafeCast<Promise<T>>().then(onFulfilled, onRejected)\n}\n", "/*\n * Copyright 2010-2018 JetBrains s.r.o.
and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license
that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.random\n\nimport
kotlin.math.pow\n\ninternal actual fun defaultPlatformRandom(): Random =\n Random(js("Math.random() *
Math.pow(2, 32)) | 0").unsafeCast<Int>())\n\nprivate val INV_2_26: Double = 2.0.pow(-26)\nprivate val
INV_2_53: Double = 2.0.pow(-53)\n\ninternal actual fun doubleFromParts(hi26: Int, low27: Int): Double =\n hi26 *
INV_2_26 + low27 * INV_2_53", "/*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\npackage kotlin.reflect\n\nimport findAssociatedObject\n\n/**\n * The
experimental marker for associated objects API.\n * Any usage of a declaration annotated with
`@ExperimentalAssociatedObjects` must be accepted either by\n * annotating that usage with the [OptIn]
annotation, e.g. `@OptIn(ExperimentalAssociatedObjects::class)`,\n * or by using the compiler argument
`-Xopt-in=kotlin.reflect.ExperimentalAssociatedObjects`.\n * \n@RequiresOptIn(level =
RequiresOptIn.Level.ERROR)\n@Retention(value = AnnotationRetention.BINARY)\npublic annotation class
ExperimentalAssociatedObjects\n\n/**\n * Makes the annotated annotation class an associated object key.\n *
An associated object key annotation should have single [KClass] parameter.\n * When applied to a class with
reference to an object declaration as an argument, it binds\n * the object to the class, making this binding
discoverable at runtime using [findAssociatedObject].\n
*\n@ExperimentalAssociatedObjects\n@Retention(AnnotationRetention.BINARY)\n@Target(AnnotationTarget.A
NNOTATION_CLASS)\npublic annotation class AssociatedObjectKey\n\n/**\n * If [T] is an
@[AssociatedObjectKey]-annotated annotation class and [this] class is annotated with @[T] (`S::class`),\n *
returns object `S`. \n * Otherwise returns `null`. \n *\n@ExperimentalAssociatedObjects\npublic inline fun <reified T :
Annotation> KClass<*>.findAssociatedObject(): Any? =\n this.findAssociatedObject(T::class)", "/*\n * Copyright
2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed
by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.js\n\nimport
getKClass\nimport kotlin.reflect.KClass\nimport kotlin.reflect.js.internal.KClassImpl\n\n/**\n * Represents the
constructor of a class. Instances of `JsClass` can be passed to JavaScript APIs that expect a constructor reference.\n
*\n@nexternal interface JsClass<T : Any> {\n /**\n * Returns the unqualified name of the class represented by
this instance.\n * \n val name: String\n }\n\n/**\n * Obtains a constructor reference for the given `KClass`. \n
*\n@nval <T : Any> KClass<T>.js: JsClass<T>\n get() = (this as KClassImpl<T>).jClass\n\n/**\n * Obtains a

```

```

`KClass` instance for the given constructor reference.\n *\/\nval <T : Any> JsClass<T>.kotlin: KClass<T>\n get()
= getKClass(this)\n", /*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n *\/\n\npackage kotlin.reflect.js.internal\n\nimport kotlin.reflect.*\n\ninternal abstract
class KClassImpl<T : Any>(\n    internal open val jClass: JsClass<T>\n) : KClass<T> {\n\n    override val
qualifiedName: String?\n    get() = TODO()\n\n    override fun equals(other: Any?): Boolean {\n        return other
is KClassImpl<*> && jClass == other.jClass\n    }\n\n    // TODO: use FQN\n    override fun hashCode(): Int =
simpleName?.hashCode() ?: 0\n\n    override fun toString(): String {\n        // TODO: use FQN\n        return "\"class
$simpleName\"\n    }\n}\n\ninternal class SimpleKClassImpl<T : Any>(jClass: JsClass<T>) :
KClassImpl<T>(jClass) {\n    override val simpleName: String? =
jClass.asDynamic().`$metadata$`.simpleName.unsafeCast<String?>()\n\n    override fun isInstance(value: Any?):
Boolean {\n        return jsIsType(value, jClass)\n    }\n}\n\ninternal class PrimitiveKClassImpl<T : Any>(\n
jClass: JsClass<T>,\n    private val givenSimpleName: String,\n    private val isInstanceFunction: (Any?) ->
Boolean\n) : KClassImpl<T>(jClass) {\n    override fun equals(other: Any?): Boolean {\n        if (other lis
PrimitiveKClassImpl<*>) return false\n        return super.equals(other) && givenSimpleName ==
other.givenSimpleName\n    }\n\n    override val simpleName: String? get() = givenSimpleName\n\n    override fun
isInstance(value: Any?): Boolean {\n        return isInstanceFunction(value)\n    }\n}\n\ninternal object
NothingKClassImpl : KClassImpl<Nothing>(js("Object")) {\n    override val simpleName: String =
\"Nothing\"\n\n    override fun isInstance(value: Any?): Boolean = false\n\n    override val jClass:
JsClass<Nothing>\n        get() = throw UnsupportedOperationException(\"There's no native JS class for Nothing
type\")\n\n    override fun equals(other: Any?): Boolean = other === this\n\n    override fun hashCode(): Int =
0\n}\n\ninternal class ErrorKClass : KClass<Nothing> {\n    override val simpleName: String? get() =
error(\"Unknown simpleName for ErrorKClass\")\n\n    override val qualifiedName: String? get() = error(\"Unknown
qualifiedName for ErrorKClass\")\n\n    override fun isInstance(value: Any?): Boolean = error(\"Can't check
isInstance on ErrorKClass\")\n\n    override fun equals(other: Any?): Boolean = other === this\n\n    override fun
hashCode(): Int = 0\n}"/,*\n * Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n *\/\n\npackage kotlin.reflect\n\ninternal actual inline val
KClass<*>.qualifiedOrSimpleName: String?\n    get() = simpleName", /*\n * Copyright 2010-2018 JetBrains s.r.o.
and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license
that can be found in the license/LICENSE.txt file.\n *\/\n\n// a package is omitted to get declarations directly under
the module\n\n// TODO: Remove once JsReflectionAPICallChecker supports more reflection
types\n@file:Suppress(\"Unsupported\")\n\nimport kotlin.reflect.*\n\nimport
kotlin.reflect.js.internal.*\n\n@JsName(\"createKType\")\n\ninternal fun createKType(\n    classifier: KClassifier,\n
arguments: Array<KTypeProjection>,\n    isMarkedNullable: Boolean\n) =\n    KTypeImpl(classifier,
arguments.asList(), isMarkedNullable)\n\n@JsName(\"createDynamicKType\")\n\ninternal fun
createDynamicKType(): KType = DynamicKType\n\n@JsName(\"markKTypeNullable\")\n\ninternal fun
markKTypeNullable(kType: KType) = KTypeImpl(kType.classifier!!, kType.arguments,
true)\n\n@JsName(\"createKTypeParameter\")\n\ninternal fun createKTypeParameter(\n    name: String,\n
upperBounds: Array<KType>,\n    variance: String\n): KTypeParameter {\n    val kVariance = when (variance) {\n
        \"in\" -> KVariance.IN\n        \"out\" -> KVariance.OUT\n        else -> KVariance.INVARIANT\n    }\n\n    return
KTypeParameterImpl(name, upperBounds.asList(), kVariance,
false)\n}\n\n@JsName(\"getStarKTypeProjection\")\n\ninternal fun getStarKTypeProjection(): KTypeProjection =\n
KTypeProjection.STAR\n\n@JsName(\"createCovariantKTypeProjection\")\n\ninternal fun
createCovariantKTypeProjection(type: KType): KTypeProjection =\n
KTypeProjection.covariant(type)\n\n@JsName(\"createInvariantKTypeProjection\")\n\ninternal fun
createInvariantKTypeProjection(type: KType): KTypeProjection =\n
KTypeProjection.invariant(type)\n\n@JsName(\"createContravariantKTypeProjection\")\n\ninternal fun

```

```

createContravariantKTypeProjection(type: KType): KTypeProjection =\n
KTypeProjection.contravariant(type)\n", "/*\n * Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\npackage kotlin.reflect.js.internal\n\nimport kotlin.reflect.*\n\ninternal class
KTypeImpl(\n    override val classifier: KClassifier,\n    override val arguments: List<KTypeProjection>,\n    override val isMarkedNullable: Boolean\n) : KType {\n    override fun equals(other: Any?): Boolean =\n        other
is KTypeImpl &&\n            classifier == other.classifier && arguments == other.arguments &&
isMarkedNullable == other.isMarkedNullable\n\n    override fun hashCode(): Int =\n        (classifier.hashCode() * 31
+ arguments.hashCode()) * 31 + isMarkedNullable.hashCode()\n\n    override fun toString(): String {\n        val
kClass = (classifier as? KClass<*>)\n        val classifierName = when {\n            kClass == null ->
classifier.toString()\n            kClass.simpleName != null -> kClass.simpleName\n            else -> "(non-denotable
type)"\n        }\n        val args =\n            if (arguments.isEmpty()) ""\n            else arguments.joinToString(", ",
"<", ">") { it.asString() }\n        val nullable = if (isMarkedNullable) "?" else ""\n        return classifierName
+ args + nullable\n    }\n\n    // TODO: this should be the implementation of KTypeProjection.toString, see KT-
30071\n    private fun KTypeProjection.asString(): String {\n        if (variance == null) return "*" \n        return
variance.prefixString() + type.toString()\n    }\n\n    \n\ninternal object DynamicKType : KType {\n    override val
classifier: KClassifier? = null\n    override val arguments: List<KTypeProjection> = emptyList()\n    override val
isMarkedNullable: Boolean = false\n    override fun toString(): String = "dynamic"\n\n    \n\ninternal fun
KVariance.prefixString() =\n    when (this) {\n        KVariance.INVARIANT -> ""\n        KVariance.IN -> "in "\n
KVariance.OUT -> "out "\n    }\n", "/*\n * Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\npackage kotlin.reflect.js.internal\n\nimport kotlin.reflect.*\n\ninternal data class
KTypeParameterImpl(\n    override val name: String,\n    override val upperBounds: List<KType>,\n    override val
variance: KVariance,\n    override val isReified: Boolean\n) : KTypeParameter {\n    override fun toString(): String
= name\n", "/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use
of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*/\n\npackage kotlin.reflect.js.internal\n\nimport kotlin.js.JsClass\n\n@JsName("PrimitiveClasses")\n\ninternal
object PrimitiveClasses {\n    @JsName("anyClass")\n    val anyClass =
PrimitiveKClassImpl(js("Object").unsafeCast<JsClass<Any>>(), "Any", { it is Any })\n\n    @JsName("numberClass")\n    val numberClass =
PrimitiveKClassImpl(js("Number").unsafeCast<JsClass<Number>>(), "Number", { it is Number })\n\n    @JsName("nothingClass")\n    val nothingClass = NothingKClassImpl\n\n    @JsName("booleanClass")\n    val
booleanClass = PrimitiveKClassImpl(js("Boolean").unsafeCast<JsClass<Boolean>>(), "Boolean", { it is Boolean
})\n\n    @JsName("byteClass")\n    val byteClass =
PrimitiveKClassImpl(js("Number").unsafeCast<JsClass<Byte>>(), "Byte", { it is Byte })\n\n    @JsName("shortClass")\n    val shortClass = PrimitiveKClassImpl(js("Number").unsafeCast<JsClass<Short>>(),
"Short", { it is Short })\n\n    @JsName("intClass")\n    val intClass =
PrimitiveKClassImpl(js("Number").unsafeCast<JsClass<Int>>(), "Int", { it is Int })\n\n    @JsName("floatClass")\n    val floatClass = PrimitiveKClassImpl(js("Number").unsafeCast<JsClass<Float>>(),
"Float", { it is Float })\n\n    @JsName("doubleClass")\n    val doubleClass =
PrimitiveKClassImpl(js("Number").unsafeCast<JsClass<Double>>(), "Double", { it is Double })\n\n    @JsName("arrayClass")\n    val arrayClass =
PrimitiveKClassImpl(js("Array").unsafeCast<JsClass<Array<*>>>(), "Array", { it is Array<*> })\n\n    @JsName("stringClass")\n    val stringClass = PrimitiveKClassImpl(js("String").unsafeCast<JsClass<String>>(),
"String", { it is String })\n\n    @JsName("throwableClass")\n    val throwableClass =
PrimitiveKClassImpl(js("Error").unsafeCast<JsClass<Throwable>>(), "Throwable", { it is Throwable })\n\n    @JsName("booleanArrayClass")\n    val booleanArrayClass =
PrimitiveKClassImpl(js("Array").unsafeCast<JsClass<BooleanArray>>(), "BooleanArray", { it is BooleanArray

```

```

})\n\n @JsName("charArrayClass")\n val charArrayClass =
PrimitiveKClassImpl(js("Uint16Array").unsafeCast<JsClass<CharArray>>(), "CharArray", { it is CharArray
})\n\n @JsName("byteArrayClass")\n val byteArrayClass =
PrimitiveKClassImpl(js("Int8Array").unsafeCast<JsClass<ByteArray>>(), "ByteArray", { it is ByteArray })\n\n
@JsName("shortArrayClass")\n val shortArrayClass =
PrimitiveKClassImpl(js("Int16Array").unsafeCast<JsClass<ShortArray>>(), "ShortArray", { it is ShortArray
})\n\n @JsName("intArrayClass")\n val intArrayClass =
PrimitiveKClassImpl(js("Int32Array").unsafeCast<JsClass<IntArray>>(), "IntArray", { it is IntArray })\n\n
@JsName("longArrayClass")\n val longArrayClass =
PrimitiveKClassImpl(js("Array").unsafeCast<JsClass<LongArray>>(), "LongArray", { it is LongArray })\n\n
@JsName("floatArrayClass")\n val floatArrayClass =
PrimitiveKClassImpl(js("Float32Array").unsafeCast<JsClass<FloatArray>>(), "FloatArray", { it is FloatArray
})\n\n @JsName("doubleArrayClass")\n val doubleArrayClass =
PrimitiveKClassImpl(js("Float64Array").unsafeCast<JsClass<DoubleArray>>(), "DoubleArray", { it is
DoubleArray })\n\n @JsName("functionClass")\n fun functionClass(arity: Int): KClassImpl<Any> {\n
return functionClasses.get(arity) ?: run {\n    val result =
PrimitiveKClassImpl(js("Function").unsafeCast<JsClass<Any>>(), "Function$arity",\n    { jsTypeOf(it) === "function" && it.asDynamic().length === arity })\n    functionClasses.asDynamic()[arity]
= result\n    result\n    }\n    }\n\nprivate val functionClasses =
arrayOfNulls<KClassImpl<Any>>(0), "/*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\n a package is omitted to get declarations directly under the module\n\nimport
kotlin.reflect.*\nimport kotlin.reflect.js.internal.*\n\n@JsName("getKClass")\ninternal fun <T : Any>
getKClass(jClass: Any /* JsClass<T> | Array<JsClass<T>> */): KClass<T> {\n    return if
(js("Array").isArray(jClass)) {\n        getKClassM(jClass.unsafeCast<Array<JsClass<T>>>())\n    } else {\n
getKClass1(jClass.unsafeCast<JsClass<T>>())\n    }\n\n@JsName("getKClassM")\ninternal fun <T : Any>
getKClassM(jClasses: Array<JsClass<T>>): KClass<T> = when (jClasses.size) {\n    1 ->
getKClass1(jClasses[0])\n    0 -> NothingKClassImpl.unsafeCast<KClass<T>>()\n    else ->
ErrorKClass().unsafeCast<KClass<T>>()\n\n@JsName("getKClassFromExpression")\ninternal fun <T : Any>
getKClassFromExpression(e: T): KClass<T> =\n    when (jsTypeOf(e)) {\n        "string" ->
PrimitiveClasses.stringClass\n        "number" -> if (jsBitwiseOr(e, 0).asDynamic() === e)
PrimitiveClasses.intClass else PrimitiveClasses.doubleClass\n        "boolean" -> PrimitiveClasses.booleanClass\n
"function" -> PrimitiveClasses.functionClass(e.asDynamic().length)\n        else -> {\n            when {\n                e
is BooleanArray -> PrimitiveClasses.booleanArrayClass\n                e is CharArray ->
PrimitiveClasses.charArrayClass\n                e is ByteArray -> PrimitiveClasses.byteArrayClass\n                e is
ShortArray -> PrimitiveClasses.shortArrayClass\n                e is IntArray -> PrimitiveClasses.intArrayClass\n
                e is LongArray -> PrimitiveClasses.longArrayClass\n                e is FloatArray ->
PrimitiveClasses.floatArrayClass\n                e is DoubleArray -> PrimitiveClasses.doubleArrayClass\n                e is
KClass<*> -> KClass::class\n                e is Array<*> -> PrimitiveClasses.arrayClass\n                else -> {\n
                    val constructor = js("Object").getPrototypeOf(e).constructor\n                    when {\n                        constructor
=== js("Object") -> PrimitiveClasses.anyClass\n                        constructor === js("Error") ->
PrimitiveClasses.throwableClass\n                        else -> {\n                            val jsClass: JsClass<T> =
constructor\n                            getKClass1(jsClass)\n                            }\n                            }\n                            }\n
                        }\n                    }\n                }\n            }\n        }\n    }\n\n@JsName("getKClass1")\ninternal fun <T : Any> getKClass1(jClass:
JsClass<T>): KClass<T> {\n    if (jClass === js("String")) return
PrimitiveClasses.stringClass.unsafeCast<KClass<T>>()\n\n    val metadata = jClass.asDynamic().`$metadata$\`\n\n
return if (metadata != null) {\n        if (metadata.`$kClass$\` == null) {\n            val kClass =
SimpleKClassImpl(jClass)\n            metadata.`$kClass$\` = kClass\n            kClass\n        } else {\n

```

```

metadata.`$kClass`$`n    }`n } else {`n    SimpleKClassImpl(jClass)`n    }`n}`,/*`n * Copyright 2010-2018
JetBrains s.r.o. and Kotlin Programming Language contributors.`n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.`n */`n`npackage kotlin.js`n`n/*`n * Exposes
the JavaScript [RegExp
object](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/RegExp) to Kotlin.`n
*`n@Suppress("NOT_DOCUMENTED")`n`npublic external class RegExp(pattern: String, flags: String? =
definedExternally) {`n`n    public fun test(str: String): Boolean`n`n    public fun exec(str: String): RegExpMatch?`n`n
    public override fun toString(): String`n`n    /*`n    * The lastIndex is a read/write integer property of regular
expressions that specifies the index at which to start the next match.`n    *`n    public var lastIndex: Int`n`n    public
val global: Boolean`n`n    public val ignoreCase: Boolean`n`n    public val multiline: Boolean`n`n}`n`n/*`n * Resets the
regular expression so that subsequent [RegExp.test] and [RegExp.exec] calls will match starting with the beginning
of the input string.`n */`n`npublic fun RegExp.reset() {`n    lastIndex = 0`n`n}`n`n// TODO: Inherit from array or
introduce asArray() extension`n`n/*`n * Represents the return value of [RegExp.exec].`n
*`n@Suppress("NOT_DOCUMENTED")`n`npublic external interface RegExpMatch {`n`n    public val index: Int`n`n
    public val input: String`n`n    public val length: Int`n`n}`n`n/*`n * Returns the entire text matched by [RegExp.exec] if
the [index] parameter is 0, or the text matched by the capturing parenthesis`n * at the given index.`n */`n`npublic inline
operator fun RegExpMatch.get(index: Int): String? = asDynamic()[index]`n`n/*`n * Converts the result of
[RegExp.exec] to an array where the first element contains the entire matched text and each subsequent`n * element
is the text matched by each capturing parenthesis.`n */`n`npublic inline fun RegExpMatch.asArray(): Array<out
String?> = unsafeCast<Array<out String?>>()`n`n}`,/*`n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin
Programming Language contributors.`n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.`n */`n`npackage kotlin.sequences`n`ninternal actual class
ConstrainedOnceSequence<T> actual constructor(sequence: Sequence<T>) : Sequence<T> {`n`n    private var
sequenceRef: Sequence<T>? = sequence`n`n    actual override fun iterator(): Iterator<T> {`n`n        val sequence =
sequenceRef ?: throw IllegalStateException("This sequence can be consumed only once.")`n`n        sequenceRef =
null`n`n        return sequence.iterator()`n`n    }`n`n}`,/*`n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin
Programming Language contributors.`n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.`n */`n`npackage kotlin.text`n`n@SinceKotlin("1.5")`n`npublic actual enum
class CharCategory(internal val value: Int, public actual val code: String) {`n`n    /*`n    * General category "Cn" in
the Unicode specification.`n    */`n`n    UNASSIGNED(0, "Cn"),`n`n    /*`n    * General category "Lu" in the
Unicode specification.`n    */`n`n    UPPERCASE_LETTER(1, "Lu"),`n`n    /*`n    * General category "Ll" in the
Unicode specification.`n    */`n`n    LOWERCASE_LETTER(2, "Ll"),`n`n    /*`n    * General category "Lt" in the
Unicode specification.`n    */`n`n    TITLECASE_LETTER(3, "Lt"),`n`n    /*`n    * General category "Lm" in the
Unicode specification.`n    */`n`n    MODIFIER_LETTER(4, "Lm"),`n`n    /*`n    * General category "Lo" in the
Unicode specification.`n    */`n`n    OTHER_LETTER(5, "Lo"),`n`n    /*`n    * General category "Mn" in the
Unicode specification.`n    */`n`n    NON_SPACING_MARK(6, "Mn"),`n`n    /*`n    * General category "Me" in
the Unicode specification.`n    */`n`n    ENCLOSING_MARK(7, "Me"),`n`n    /*`n    * General category "Mc" in
the Unicode specification.`n    */`n`n    COMBINING_SPACING_MARK(8, "Mc"),`n`n    /*`n    * General
category "Nd" in the Unicode specification.`n    */`n`n    DECIMAL_DIGIT_NUMBER(9, "Nd"),`n`n    /*`n    *
General category "Nl" in the Unicode specification.`n    */`n`n    LETTER_NUMBER(10, "Nl"),`n`n    /*`n    *
General category "No" in the Unicode specification.`n    */`n`n    OTHER_NUMBER(11, "No"),`n`n    /*`n    *
General category "Zs" in the Unicode specification.`n    */`n`n    SPACE_SEPARATOR(12, "Zs"),`n`n    /*`n    *
General category "Zl" in the Unicode specification.`n    */`n`n    LINE_SEPARATOR(13, "Zl"),`n`n    /*`n    *
General category "Zp" in the Unicode specification.`n    */`n`n    PARAGRAPH_SEPARATOR(14, "Zp"),`n`n
    /*`n    * General category "Cc" in the Unicode specification.`n    */`n`n    CONTROL(15, "Cc"),`n`n    /*`n    *
General category "Cf" in the Unicode specification.`n    */`n`n    FORMAT(16, "Cf"),`n`n    /*`n    * General
category "Co" in the Unicode specification.`n    */`n`n    PRIVATE_USE(18, "Co"),`n`n    /*`n    * General
category "Cs" in the Unicode specification.`n    */`n`n    SURROGATE(19, "Cs"),`n`n    /*`n    * General category

```

```

\Pd\ in the Unicode specification.\n */\n DASH_PUNCTUATION(20, \Pd\),\n\n /**\n * General
category \Ps\ in the Unicode specification.\n */\n START_PUNCTUATION(21, \Ps\),\n\n /**\n *
General category \Pe\ in the Unicode specification.\n */\n END_PUNCTUATION(22, \Pe\),\n\n /**\n *
General category \Pc\ in the Unicode specification.\n */\n CONNECTOR_PUNCTUATION(23, \Pc\),\n\n
/**\n * General category \Po\ in the Unicode specification.\n */\n OTHER_PUNCTUATION(24,
\Po\),\n\n /**\n * General category \Sm\ in the Unicode specification.\n */\n MATH_SYMBOL(25,
\Sm\),\n\n /**\n * General category \Sc\ in the Unicode specification.\n */\n
CURRENCY_SYMBOL(26, \Sc\),\n\n /**\n * General category \Sk\ in the Unicode specification.\n */\n
MODIFIER_SYMBOL(27, \Sk\),\n\n /**\n * General category \So\ in the Unicode specification.\n */\n
OTHER_SYMBOL(28, \So\),\n\n /**\n * General category \Pi\ in the Unicode specification.\n */\n
INITIAL_QUOTE_PUNCTUATION(29, \Pi\),\n\n /**\n * General category \Pf\ in the Unicode
specification.\n */\n FINAL_QUOTE_PUNCTUATION(30, \Pf\);\n\n /**\n * Returns `true` if [char]
character belongs to this category.\n */\n public actual operator fun contains(char: Char): Boolean =
char.getCategoryValue() == this.value\n\n companion object {\n    internal fun valueOf(category: Int):
CharCategory =\n        when (category) {\n            in 0..16 -> values()[category]\n            in 18..30 ->
values()[category - 1]\n            else -> throw IllegalArgumentException(\nCategory #\$category is not defined.\n)\n        }\n    }\n\n"/\n\n * Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming Language contributors.\n
* Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*/\n\npackage kotlin.text\n\n/**\n * The exception thrown when a character encoding or decoding error occurs.\n
*/\n\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic actual open class
CharacterCodingException(message: String?) : Exception(message) {\n    actual constructor() : this(null)\n\n"/\n\n
* Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code
is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage
kotlin.text\n\n/**\n * A mutable sequence of characters.\n */\n\n * String builder can be used to efficiently perform
multiple string manipulation operations.\n */\n\npublic actual class StringBuilder actual constructor(content: String) :
Appendable, CharSequence {\n    /**\n     * Constructs an empty string builder with the specified initial [capacity].\n
*/\n     * In Kotlin/JS implementation of StringBuilder the initial capacity has no effect on the further performance
of operations.\n     */\n     actual constructor(capacity: Int) : this() {\n    }\n\n    /**\n     * Constructs a string builder that
contains the same characters as the specified [content] char sequence.\n     */\n     actual constructor(content:
CharSequence) : this(content.toString()) {\n    }\n\n    /**\n     * Constructs an empty string builder.\n     */\n     actual constructor() :
this("")\n\n    private var string: String = if (content != undefined) content else ""\n\n     actual override val
length: Int\n     get() = string.asDynamic().length\n\n     actual override fun get(index: Int): Char =\n
string.getOrElse(index) { throw IndexOutOfBoundsException(\nindex: \$index, length: \$length)\n}\n\n     actual
override fun subSequence(startIndex: Int, endIndex: Int): CharSequence = string.substring(startIndex, endIndex)\n\n
     actual override fun append(value: Char): StringBuilder {\n     string += value\n     return this\n    }\n\n     actual
override fun append(value: CharSequence?): StringBuilder {\n     string += value.toString()\n     return this\n    }\n\n
     actual override fun append(value: CharSequence?, startIndex: Int, endIndex: Int): StringBuilder =\n
this.appendRange(value?: \nnull", startIndex, endIndex)\n\n     /**\n     * Reverses the contents of this string builder
and returns this instance.\n     */\n     * Surrogate pairs included in this string builder are treated as single
characters.\n     * Therefore, the order of the high-low surrogates is never reversed.\n     */\n     * Note that the reverse
operation may produce new surrogate pairs that were unpaired low-surrogates and high-surrogates before the
operation.\n     * For example, reversing "\uDC00\uD800" produces "\uD800\uDC00" which is a valid
surrogate pair.\n     */\n     actual fun reverse(): StringBuilder {\n     var reversed = ""\n     var index =
string.length - 1\n     while (index >= 0) {\n     val low = string[index--]\n     if (low.isLowSurrogate()) &&
index >= 0) {\n     val high = string[index--]\n     if (high.isHighSurrogate()) {\n     reversed =
reversed + high + low\n     } else {\n     reversed = reversed + low + high\n     }\n     }\n     else {\n     reversed += low\n     }\n     }\n     string = reversed\n     return this\n    }\n\n     /**\n     *
Appends the string representation of the specified object [value] to this string builder and returns this instance.\n
*/\n\n     actual fun append(value: Any?): StringBuilder {\n     string += value.toString()\n     return this\n    }\n\n
}

```

```

*\n * The overall effect is exactly as if the [value] were converted to a string by the `value.toString()` method,\n
* and then that string was appended to this string builder.\n */\n actual fun append(value: Any?): StringBuilder\n
{\n     string += value.toString()\n     return this\n }\n\n /**\n * Appends the string representation of the\n
specified boolean [value] to this string builder and returns this instance.\n */\n * The overall effect is exactly as\n
if the [value] were converted to a string by the `value.toString()` method,\n * and then that string was appended to\n
this string builder.\n */\n @SinceKotlin("1.3")\n actual fun append(value: Boolean): StringBuilder {\n\n
string += value\n     return this\n }\n\n /**\n * Appends characters in the specified character array [value] to\n
this string builder and returns this instance.\n */\n * Characters are appended in order, starting at the index 0.\n
*/\n @SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n actual fun append(value:\n
CharArray): StringBuilder {\n     string += value.concatToString()\n     return this\n }\n\n\n
@Deprecated("Provided for binary compatibility.", level = DeprecationLevel.HIDDEN)\n fun append(value:\n
String): StringBuilder = append(value)\n\n /**\n * Appends the specified string [value] to this string builder and\n
returns this instance.\n */\n * If [value] is `null`, then the four characters `"null"` are appended.\n */\n\n
@SinceKotlin("1.3")\n actual fun append(value: String?): StringBuilder {\n     this.string += value ?: "null"\n\n
return this\n }\n\n /**\n * Returns the current capacity of this string builder.\n */\n * The capacity is the\n
maximum length this string builder can have before an allocation occurs.\n */\n * In Kotlin/JS implementation\n
of StringBuilder the value returned from this method may not indicate the actual size of the backing storage.\n
*/\n @SinceKotlin("1.3")\n // @ExperimentalStdlibApi\n @Deprecated("Obtaining String Builder capacity is\n
not supported in JS and common code.", level = DeprecationLevel.ERROR)\n actual fun capacity(): Int =\n
length\n\n /**\n * Ensures that the capacity of this string builder is at least equal to the specified\n
[minimumCapacity].\n */\n * If the current capacity is less than the [minimumCapacity], a new backing storage\n
is allocated with greater capacity.\n */\n * Otherwise, this method takes no action and simply returns.\n */\n * In\n
Kotlin/JS implementation of String Builder the size of the backing storage is not extended to comply the given\n
[minimumCapacity],\n * thus calling this method has no effect on the further performance of operations.\n */\n\n
@SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n actual fun\n
ensureCapacity(minimumCapacity: Int) {\n }\n\n /**\n * Returns the index within this string builder of the\n
first occurrence of the specified [string].\n */\n * Returns -1 if the specified [string] does not occur in this\n
string builder.\n */\n @SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n actual\n
fun indexOf(string: String): Int = this.string.asDynamic().indexOf(string)\n\n /**\n * Returns the index within\n
this string builder of the first occurrence of the specified [string],\n * starting at the specified [startIndex].\n */\n\n
* Returns -1 if the specified [string] does not occur in this string builder starting at the specified [startIndex].\n
*/\n @SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n actual fun indexOf(string:\n
String, startIndex: Int): Int = this.string.asDynamic().indexOf(string, startIndex)\n\n /**\n * Returns the index\n
within this string builder of the last occurrence of the specified [string].\n * The last occurrence of empty string\n
`""` is considered to be at the index equal to `this.length`.\n */\n * Returns -1 if the specified [string] does not\n
occur in this string builder.\n */\n @SinceKotlin("1.4")\n\n
@WasExperimental(ExperimentalStdlibApi::class)\n actual fun lastIndexOf(string: String): Int =\n
this.string.asDynamic().lastIndexOf(string)\n\n /**\n * Returns the index within this string builder of the last\n
occurrence of the specified [string],\n * starting from the specified [startIndex] toward the beginning.\n */\n * \n
Returns -1 if the specified [string] does not occur in this string builder starting at the specified [startIndex].\n
*/\n @SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n actual fun\n
lastIndexOf(string: String, startIndex: Int): Int {\n     if (string.isEmpty() && startIndex < 0) return -1\n     return\n
this.string.asDynamic().lastIndexOf(string, startIndex)\n }\n\n /**\n * Inserts the string representation of the\n
specified boolean [value] into this string builder at the specified [index] and returns this instance.\n */\n * The\n
overall effect is exactly as if the [value] were converted to a string by the `value.toString()` method,\n * and then\n
that string was inserted into this string builder at the specified [index].\n */\n * @throws\n
IndexOutOfBoundsException if [index] is less than zero or greater than the length of this string builder.\n */\n\n
@SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n actual fun insert(index: Int, value:

```

```

Boolean): StringBuilder {\n    AbstractList.checkPositionIndex(index, length)\n\n    string = string.substring(0,
index) + value + string.substring(index)\n    return this\n }\n\n /**\n  * Inserts the specified character [value]
into this string builder at the specified [index] and returns this instance.\n  *\n  * @throws
IndexOutOfBoundsException if [index] is less than zero or greater than the length of this string builder.\n  */\n
@SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n actual fun insert(index: Int, value:
Char): StringBuilder {\n    AbstractList.checkPositionIndex(index, length)\n\n    string = string.substring(0,
index) + value + string.substring(index)\n    return this\n }\n\n /**\n  * Inserts characters in the specified
character array [value] into this string builder at the specified [index] and returns this instance.\n  *\n  * The
inserted characters go in same order as in the [value] character array, starting at [index].\n  *\n  * @throws
IndexOutOfBoundsException if [index] is less than zero or greater than the length of this string builder.\n  */\n
@SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n actual fun insert(index: Int, value:
CharArray): StringBuilder {\n    AbstractList.checkPositionIndex(index, length)\n\n    string =
string.substring(0, index) + value.concatToString() + string.substring(index)\n    return this\n }\n\n /**\n  *
Inserts characters in the specified character sequence [value] into this string builder at the specified [index] and
returns this instance.\n  *\n  * The inserted characters go in the same order as in the [value] character sequence,
starting at [index].\n  *\n  * @param index the position in this string builder to insert at.\n  * @param value the
character sequence from which characters are inserted. If [value] is `null`, then the four characters `"\null"` are
inserted.\n  *\n  * @throws IndexOutOfBoundsException if [index] is less than zero or greater than the length of
this string builder.\n  */\n @SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n actual fun
insert(index: Int, value: CharSequence?): StringBuilder {\n    AbstractList.checkPositionIndex(index,
length)\n\n    string = string.substring(0, index) + value.toString() + string.substring(index)\n    return this\n
}\n\n /**\n  * Inserts the string representation of the specified object [value] into this string builder at the
specified [index] and returns this instance.\n  *\n  * The overall effect is exactly as if the [value] were converted
to a string by the `value.toString()` method,\n  * and then that string was inserted into this string builder at the
specified [index].\n  *\n  * @throws IndexOutOfBoundsException if [index] is less than zero or greater than the
length of this string builder.\n  */\n @SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n
actual fun insert(index: Int, value: Any?): StringBuilder {\n    AbstractList.checkPositionIndex(index, length)\n\n
string = string.substring(0, index) + value.toString() +
string.substring(index)\n    return this\n }\n\n @Deprecated("Provided for binary compatibility.", level =
DeprecationLevel.HIDDEN)\n fun insert(index: Int, value: String): StringBuilder = insert(index, value)\n\n /**\n
* Inserts the string [value] into this string builder at the specified [index] and returns this instance.\n  *\n  * If
[value] is `null`, then the four characters `"\null"` are inserted.\n  *\n  * @throws IndexOutOfBoundsException
if [index] is less than zero or greater than the length of this string builder.\n  */\n @SinceKotlin("1.4")\n
@WasExperimental(ExperimentalStdlibApi::class)\n actual fun insert(index: Int, value: String?): StringBuilder
{\n    AbstractList.checkPositionIndex(index, length)\n\n    val toInsert = value ?: "\null"\n    this.string =
this.string.substring(0, index) + toInsert + this.string.substring(index)\n    return this\n }\n\n /**\n  * Sets
the length of this string builder to the specified [newLength].\n  *\n  * If the [newLength] is less than the current
length, it is changed to the specified [newLength].\n  * Otherwise, null characters `"\u0000"` are appended to
this string builder until its length is less than the [newLength].\n  *\n  * Note that in Kotlin/JS [set] operator
function has non-constant execution time complexity.\n  * Therefore, increasing length of this string builder and
then updating each character by index may slow down your program.\n  *\n  * @throws
IndexOutOfBoundsException or [IllegalArgumentException] if [newLength] is less than zero.\n  */\n @SinceKotlin("1.4")\n
@WasExperimental(ExperimentalStdlibApi::class)\n actual fun setLength(newLength:
Int) {\n    if (newLength < 0) {\n        throw IllegalArgumentException("Negative new length:
$newLength.")\n    }\n\n    if (newLength <= length) {\n        string = string.substring(0, newLength)\n    }
else {\n        for (i in length until newLength) {\n            string += "\u0000"\n        }\n    }\n }\n\n /**\n
* Returns a new [String] that contains characters in this string builder at [startIndex] (inclusive) and up to the
[length] (exclusive).\n  *\n  * @throws IndexOutOfBoundsException if [startIndex] is less than zero or greater

```

```

than the length of this string builder.\n    */\n    @SinceKotlin("1.4")\n    @WasExperimental(ExperimentalStdlibApi::class)\n    actual fun substring(startIndex: Int): String {\n    AbstractList.checkPositionIndex(startIndex, length)\n    return string.substring(startIndex)\n    }\n\n    /**\n    * Returns a new [String] that contains characters in this string builder at [startIndex] (inclusive) and up to the\n    [endIndex] (exclusive).\n    *\n    * @throws IndexOutOfBoundsException or [IllegalArgumentException] when\n    [startIndex] or [endIndex] is out of range of this string builder indices or when `startIndex > endIndex`.\n    */\n    @SinceKotlin("1.4")\n    @WasExperimental(ExperimentalStdlibApi::class)\n    actual fun substring(startIndex:\n    Int, endIndex: Int): String {\n    AbstractList.checkBoundsIndexes(startIndex, endIndex, length)\n    return\n    string.substring(startIndex, endIndex)\n    }\n\n    /**\n    * Attempts to reduce storage used for this string builder.\n    *\n    * If the backing storage of this string builder is larger than necessary to hold its current contents,\n    * then\n    it may be resized to become more space efficient.\n    *\n    * Calling this method may, but is not required to, affect the\n    value of the [capacity] property.\n    *\n    * In Kotlin/JS implementation of StringBuilder the size of the backing\n    storage is always equal to the length of the string builder.\n    */\n    @SinceKotlin("1.4")\n    @WasExperimental(ExperimentalStdlibApi::class)\n    actual fun trimToSize() {\n    }\n\n    override fun toString():\n    String = string\n\n    /**\n    * Clears the content of this string builder making it empty and returns this instance.\n    *\n    * @sample samples.text.Strings.clearStringBuilder\n    */\n    @SinceKotlin("1.3")\n    public fun clear():\n    StringBuilder {\n    string = ""\n    return this\n    }\n\n    /**\n    * Sets the character at the specified [index]\n    to the specified [value].\n    *\n    * @throws IndexOutOfBoundsException if [index] is out of bounds of this string\n    builder.\n    */\n    @SinceKotlin("1.4")\n    @WasExperimental(ExperimentalStdlibApi::class)\n    public\n    operator fun set(index: Int, value: Char) {\n    AbstractList.checkElementIndex(index, length)\n    string =\n    string.substring(0, index) + value + string.substring(index + 1)\n    }\n\n    /**\n    * Replaces characters in the\n    specified range of this string builder with characters in the specified string [value] and returns this instance.\n    *\n    * @param startIndex the beginning (inclusive) of the range to replace.\n    * @param endIndex the end (exclusive)\n    of the range to replace.\n    * @param value the string to replace with.\n    *\n    * @throws\n    IndexOutOfBoundsException or [IllegalArgumentException] if [startIndex] is less than zero, greater than the length\n    of this string builder, or `startIndex > endIndex`.\n    */\n    @SinceKotlin("1.4")\n    @WasExperimental(ExperimentalStdlibApi::class)\n    public fun setRange(startIndex: Int, endIndex: Int, value:\n    String): StringBuilder {\n    checkReplaceRange(startIndex, endIndex, length)\n    this.string =\n    this.string.substring(0, startIndex) + value + this.string.substring(endIndex)\n    return this\n    }\n\n    private fun\n    checkReplaceRange(startIndex: Int, endIndex: Int, length: Int) {\n    if (startIndex < 0 || startIndex > length) {\n    throw IndexOutOfBoundsException("startIndex: $startIndex, length: $length")\n    }\n    if (startIndex >\n    endIndex) {\n    throw IllegalArgumentException("startIndex($startIndex) > endIndex($endIndex)")\n    }\n    }\n\n    /**\n    * Removes the character at the specified [index] from this string builder and returns this instance.\n    *\n    * If the `Char` at the specified [index] is part of a supplementary code point, this method does not remove\n    the entire supplementary character.\n    *\n    * @param index the index of `Char` to remove.\n    *\n    * @throws\n    IndexOutOfBoundsException if [index] is out of bounds of this string builder.\n    */\n    @SinceKotlin("1.4")\n    @WasExperimental(ExperimentalStdlibApi::class)\n    public fun deleteAt(index: Int): StringBuilder {\n    AbstractList.checkElementIndex(index, length)\n    string = string.substring(0, index) + string.substring(index +\n    1)\n    return this\n    }\n\n    /**\n    * Removes characters in the specified range from this string builder and\n    returns this instance.\n    *\n    * @param startIndex the beginning (inclusive) of the range to remove.\n    *\n    * @param endIndex the end (exclusive) of the range to remove.\n    *\n    * @throws IndexOutOfBoundsException\n    or [IllegalArgumentException] when [startIndex] is out of range of this string builder indices or when `startIndex >\n    endIndex`.\n    */\n    @SinceKotlin("1.4")\n    @WasExperimental(ExperimentalStdlibApi::class)\n    public fun\n    deleteRange(startIndex: Int, endIndex: Int): StringBuilder {\n    checkReplaceRange(startIndex, endIndex,\n    length)\n    string = string.substring(0, startIndex) + string.substring(endIndex)\n    return this\n    }\n\n    /**\n    * Copies characters from this string builder into the [destination] character array.\n    *\n    * @param\n    destination the array to copy to.\n    * @param destinationOffset the position in the array to copy to, 0 by default.\n    *\n    * @param\n    startIndex the beginning (inclusive) of the range to copy, 0 by default.\n    * @param endIndex the end

```

```

(exclusive) of the range to copy, length of this string builder by default.\n * \n * @throws
IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this
string builder indices or when `startIndex > endIndex`.\n * @throws IndexOutOfBoundsException when the
subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],\n * or when that index
is out of the [destination] array indices range.\n * \n * @SinceKotlin("1.4")\n
@WasExperimental(ExperimentalStdlibApi::class)\n public fun toCharArray(destination: CharArray,
destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = this.length) {\n
AbstractList.checkBoundsIndexes(startIndex, endIndex, length)\n
AbstractList.checkBoundsIndexes(destinationOffset, destinationOffset + endIndex - startIndex, destination.size)\n\n
    var dstIndex = destinationOffset\n    for (index in startIndex until endIndex) {\n        destination[dstIndex++]
= string[index]\n    }\n    }\n\n /**\n * Appends characters in a subarray of the specified character array
[value] to this string builder and returns this instance.\n * \n * Characters are appended in order, starting at
specified [startIndex].\n * \n * @param value the array from which characters are appended.\n * @param
startIndex the beginning (inclusive) of the subarray to append.\n * @param endIndex the end (exclusive) of the
subarray to append.\n * \n * @throws IndexOutOfBoundsException or [IllegalArgumentException] when
[startIndex] or [endIndex] is out of range of the [value] array indices or when `startIndex > endIndex`.\n * \n
* @SinceKotlin("1.4")\n * @WasExperimental(ExperimentalStdlibApi::class)\n public fun appendRange(value:
CharArray, startIndex: Int, endIndex: Int): StringBuilder {\n    string += value.concatToString(startIndex,
endIndex)\n    return this\n }\n\n /**\n * Appends a subsequence of the specified character sequence [value]
to this string builder and returns this instance.\n * \n * @param value the character sequence from which a
subsequence is appended.\n * @param startIndex the beginning (inclusive) of the subsequence to append.\n *
* @param endIndex the end (exclusive) of the subsequence to append.\n * \n * @throws
IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of the
[value] character sequence indices or when `startIndex > endIndex`.\n * \n * @SinceKotlin("1.4")\n *
@WasExperimental(ExperimentalStdlibApi::class)\n public fun appendRange(value: CharSequence, startIndex:
Int, endIndex: Int): StringBuilder {\n    val stringCsq = value.toString()\n
AbstractList.checkBoundsIndexes(startIndex, endIndex, stringCsq.length)\n\n    string +=
stringCsq.substring(startIndex, endIndex)\n    return this\n }\n\n /**\n * Inserts characters in a subarray of
the specified character array [value] into this string builder at the specified [index] and returns this instance.\n
* \n * The inserted characters go in same order as in the [value] array, starting at [index].\n * \n * @param index
the position in this string builder to insert at.\n * @param value the array from which characters are inserted.\n
* @param startIndex the beginning (inclusive) of the subarray to insert.\n * @param endIndex the end (exclusive)
of the subarray to insert.\n * \n * @throws IndexOutOfBoundsException or [IllegalArgumentException] when
[startIndex] or [endIndex] is out of range of the [value] array indices or when `startIndex > endIndex`.\n *
* @throws IndexOutOfBoundsException if [index] is less than zero or greater than the length of this string builder.\n
* \n * @SinceKotlin("1.4")\n * @WasExperimental(ExperimentalStdlibApi::class)\n public fun
insertRange(index: Int, value: CharArray, startIndex: Int, endIndex: Int): StringBuilder {\n
AbstractList.checkPositionIndex(index, this.length)\n\n    string = string.substring(0, index) +
value.concatToString(startIndex, endIndex) + string.substring(index)\n    return this\n }\n\n /**\n * Inserts
characters in a subsequence of the specified character sequence [value] into this string builder at the specified
[index] and returns this instance.\n * \n * The inserted characters go in the same order as in the [value] character
sequence, starting at [index].\n * \n * @param index the position in this string builder to insert at.\n *
* @param value the character sequence from which a subsequence is inserted.\n * @param startIndex the beginning
(inclusive) of the subsequence to insert.\n * @param endIndex the end (exclusive) of the subsequence to insert.\n
* \n * @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is
out of range of the [value] character sequence indices or when `startIndex > endIndex`.\n * @throws
IndexOutOfBoundsException if [index] is less than zero or greater than the length of this string builder.\n * \n
* @SinceKotlin("1.4")\n * @WasExperimental(ExperimentalStdlibApi::class)\n public fun insertRange(index: Int,

```

```

value: CharSequence, startIndex: Int, endIndex: Int): StringBuilder {
    AbstractList.checkPositionIndex(index,
length)
    val stringCsq = value.toString()
    AbstractList.checkBoundsIndexes(startIndex, endIndex,
stringCsq.length)
    string = string.substring(0, index) + stringCsq.substring(startIndex, endIndex) +
string.substring(index)
    return this
}

/**
 * Clears the content of this string builder making it
empty and returns this instance.
 */
@sample samples.text.Strings.clearStringBuilder
@SinceKotlin("1.3")
@Suppress("EXTENSION_SHADOWED_BY_MEMBER",
"NOTHING_TO_INLINE")
public actual inline fun StringBuilder.clear(): StringBuilder = this.clear()

/**
 * Sets the character at the specified [index] to the specified [value].
 * @throws IndexOutOfBoundsException if
[index] is out of bounds of this string builder.
 */
@SinceKotlin("1.4")
@WasExperimental(ExperimentalStdlibApi::class)
@Suppress("EXTENSION_SHA
DOWED_BY_MEMBER", "NOTHING_TO_INLINE")
public actual inline operator fun
StringBuilder.set(index: Int, value: Char) = this.set(index, value)

/**
 * Replaces characters in the specified
range of this string builder with characters in the specified string [value] and returns this instance.
 * @param
startIndex the beginning (inclusive) of the range to replace.
 * @param endIndex the end (exclusive) of the range to
replace.
 * @param value the string to replace with.
 * @throws IndexOutOfBoundsException or
[IllegalArgumentException] if [startIndex] is less than zero, greater than the length of this string builder, or
`startIndex > endIndex`.
 */
@SinceKotlin("1.4")
@WasExperimental(ExperimentalStdlibApi::class)
@Suppress("EXTENSION_SHA
DOWED_BY_MEMBER", "NOTHING_TO_INLINE")
public actual inline fun
StringBuilder.setRange(startIndex: Int, endIndex: Int, value: String): StringBuilder =
this.setRange(startIndex,
endIndex, value)

/**
 * Removes the character at the specified [index] from this string builder and returns this
instance.
 * If the `Char` at the specified [index] is part of a supplementary code point, this method does not
remove the entire supplementary character.
 * @param index the index of `Char` to remove.
 * @throws
IndexOutOfBoundsException if [index] is out of bounds of this string builder.
 */
@SinceKotlin("1.4")
@WasExperimental(ExperimentalStdlibApi::class)
@Suppress("EXTENSION_SHA
DOWED_BY_MEMBER", "NOTHING_TO_INLINE")
public actual inline fun
StringBuilder.deleteAt(index:
Int): StringBuilder = this.deleteAt(index)

/**
 * Removes characters in the specified range from this string
builder and returns this instance.
 * @param startIndex the beginning (inclusive) of the range to remove.
 * @param
endIndex the end (exclusive) of the range to remove.
 * @throws IndexOutOfBoundsException or
[IllegalArgumentException] when [startIndex] is out of range of this string builder indices or when `startIndex >
endIndex`.
 */
@SinceKotlin("1.4")
@WasExperimental(ExperimentalStdlibApi::class)
@Suppress("EXTENSION_SHA
DOWED_BY_MEMBER", "NOTHING_TO_INLINE")
public actual inline fun
StringBuilder.deleteRange(startIndex: Int, endIndex: Int): StringBuilder =
this.deleteRange(startIndex,
endIndex)

/**
 * Copies characters from this string builder into the [destination] character array.
 * @param
destination the array to copy to.
 * @param destinationOffset the position in the array to copy to, 0 by
default.
 * @param startIndex the beginning (inclusive) of the range to copy, 0 by default.
 * @param
endIndex the end (exclusive) of the range to copy, length of this string builder by default.
 * @throws
IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this
string builder indices or when `startIndex > endIndex`.
 * @throws IndexOutOfBoundsException when the
subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],
 * or when that index is
out of the [destination] array indices range.
 */
@SinceKotlin("1.4")
@WasExperimental(ExperimentalStdlibApi::class)
@Suppress("EXTENSION_SHA
DOWED_BY_MEMBER", "NOTHING_TO_INLINE",
"ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual inline fun
StringBuilder.toCharArray(destination: CharArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int =
this.length) =
this.toCharArray(destination, destinationOffset, startIndex, endIndex)

/**
 * Appends
characters in a subarray of the specified character array [value] to this string builder and returns this instance.
 */

```

Characters are appended in order, starting at specified [startIndex].\n \* @param value the array from which characters are appended.\n \* @param startIndex the beginning (inclusive) of the subarray to append.\n \* @param endIndex the end (exclusive) of the subarray to append.\n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of the [value] array indices or when `startIndex > endIndex`.\n

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@Suppress("EXTENSION_SHA  
DOWED_BY_MEMBER", "NOTHING_TO_INLINE")\npublic actual inline fun
```

```
StringBuilder.appendRange(value: CharArray, startIndex: Int, endIndex: Int): StringBuilder =\n
```

```
this.appendRange(value, startIndex, endIndex)\n\n/**\n * Appends a subsequence of the specified character  
sequence [value] to this string builder and returns this instance.\n * @param value the character sequence from  
which a subsequence is appended.\n * @param startIndex the beginning (inclusive) of the subsequence to append.\n * @param endIndex the end (exclusive) of the subsequence to append.\n * @throws  
IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of the  
[value] character sequence indices or when `startIndex > endIndex`.\n
```

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@Suppress("EXTENSION_SHA  
DOWED_BY_MEMBER", "NOTHING_TO_INLINE")\npublic actual inline fun
```

```
StringBuilder.appendRange(value: CharSequence, startIndex: Int, endIndex: Int): StringBuilder =\n
```

```
this.appendRange(value, startIndex, endIndex)\n\n/**\n * Inserts characters in a subarray of the specified character  
array [value] into this string builder at the specified [index] and returns this instance.\n * @param value the array from which characters are inserted.\n * @param startIndex the beginning  
(inclusive) of the subarray to insert.\n * @param endIndex the end (exclusive) of the subarray to insert.\n * @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of  
range of the [value] array indices or when `startIndex > endIndex`.\n * @throws IndexOutOfBoundsException if  
[index] is less than zero or greater than the length of this string builder.\n
```

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@Suppress("EXTENSION_SHA  
DOWED_BY_MEMBER", "NOTHING_TO_INLINE")\npublic actual inline fun
```

```
StringBuilder.insertRange(index: Int, value: CharArray, startIndex: Int, endIndex: Int): StringBuilder =\n
```

```
this.insertRange(index, value, startIndex, endIndex)\n\n/**\n * Inserts characters in a subsequence of the specified  
character sequence [value] into this string builder at the specified [index] and returns this instance.\n * @param value the character sequence from which a  
subsequence is inserted.\n * @param startIndex the beginning (inclusive) of the subsequence to insert.\n * @param  
endIndex the end (exclusive) of the subsequence to insert.\n * @throws IndexOutOfBoundsException or  
[IllegalArgumentException] when [startIndex] or [endIndex] is out of range of the [value] character sequence  
indices or when `startIndex > endIndex`.\n * @throws IndexOutOfBoundsException if [index] is less than zero or  
greater than the length of this string builder.\n
```

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@Suppress("EXTENSION_SHA  
DOWED_BY_MEMBER", "NOTHING_TO_INLINE")\npublic actual inline fun
```

```
StringBuilder.insertRange(index: Int, value: CharSequence, startIndex: Int, endIndex: Int): StringBuilder =\n
```

```
this.insertRange(index, value, startIndex, endIndex)\n\n"/**\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin  
Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be  
found in the license/LICENSE.txt file.\n */\n\npackage kotlin.text\n\n/**\n * Returns `true` if the content of this  
string is equal to the word `true`, ignoring case, and `false` otherwise.\n * @since 1.4 to avoid deprecation warning.\n */\n@Deprecated("Use Kotlin compiler  
1.4 to avoid deprecation warning.")\n@DeprecatedSinceKotlin(hiddenSince =
```

```
"1.4")\n@kotlin.internal.InlineOnly\npublic actual inline fun String.toBoolean(): Boolean =
```

```
this.toBoolean()\n\n/**\n * Returns `true` if this string is not `null` and its content is equal to the word `true`,  
ignoring case, and `false` otherwise.\n * There are also strict versions of the function available on non-nullable
```

String, [toBooleanStrict] and [toBooleanStrictOrNull].\n \*^@SinceKotlin("1.4")\npublic actual fun String?.toBoolean(): Boolean = this != null && this.lowercase() == "true"\n\n/\*\*\n \* Parses the string as a signed [Byte] number and returns the result.\n \* @throws NumberFormatException if the string is not a valid representation of a number.\n \*/\npublic actual fun String.toByte(): Byte = toByteOrNull() ?: numberFormatError(this)\n\n/\*\*\n \* Parses the string as a signed [Byte] number and returns the result.\n \* @throws NumberFormatException if the string is not a valid representation of a number.\n \* @throws IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n \*/\npublic actual fun String.toByte(radix: Int): Byte = toByteOrNull(radix) ?: numberFormatError(this)\n\n/\*\*\n \* Parses the string as a [Short] number and returns the result.\n \* @throws NumberFormatException if the string is not a valid representation of a number.\n \*/\npublic actual fun String.toShort(): Short = toShortOrNull() ?: numberFormatError(this)\n\n/\*\*\n \* Parses the string as a [Short] number and returns the result.\n \* @throws NumberFormatException if the string is not a valid representation of a number.\n \* @throws IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n \*/\npublic actual fun String.toShort(radix: Int): Short = toShortOrNull(radix) ?: numberFormatError(this)\n\n/\*\*\n \* Parses the string as an [Int] number and returns the result.\n \* @throws NumberFormatException if the string is not a valid representation of a number.\n \* @throws IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n \*/\npublic actual fun String.toInt(): Int = toIntOrNull() ?: numberFormatError(this)\n\n/\*\*\n \* Parses the string as an [Int] number and returns the result.\n \* @throws NumberFormatException if the string is not a valid representation of a number.\n \* @throws IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n \*/\npublic actual fun String.toInt(radix: Int): Int = toIntOrNull(radix) ?: numberFormatError(this)\n\n/\*\*\n \* Parses the string as a [Long] number and returns the result.\n \* @throws NumberFormatException if the string is not a valid representation of a number.\n \*/\npublic actual fun String.toLong(): Long = toLongOrNull() ?: numberFormatError(this)\n\n/\*\*\n \* Parses the string as a [Long] number and returns the result.\n \* @throws NumberFormatException if the string is not a valid representation of a number.\n \* @throws IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n \*/\npublic actual fun String.toLong(radix: Int): Long = toLongOrNull(radix) ?: numberFormatError(this)\n\n/\*\*\n \* Parses the string as a [Double] number and returns the result.\n \* @throws NumberFormatException if the string is not a valid representation of a number.\n \*/\npublic actual fun String.toDouble(): Double = +(this.asDynamic()).unsafeCast<Double>().also {\n if (it.isNaN() && !this.isNaN() || it == 0.0 && this.isBlank())\n numberFormatError(this)\n}\n\n/\*\*\n \* Parses the string as a [Float] number and returns the result.\n \* @throws NumberFormatException if the string is not a valid representation of a number.\n \*/\n@kotlin.internal.InlineOnly\npublic actual inline fun String.toFloat(): Float = toDouble().unsafeCast<Float>()\n\n/\*\*\n \* Parses the string as a [Double] number and returns the result\n \* or `null` if the string is not a valid representation of a number.\n \*/\npublic actual fun String.toDoubleOrNull(): Double? = +(this.asDynamic()).unsafeCast<Double>().takeIf {\n !(it.isNaN() && !this.isNaN() || it == 0.0 && this.isBlank())\n}\n\n/\*\*\n \* Parses the string as a [Float] number and returns the result\n \* or `null` if the string is not a valid representation of a number.\n \*/\n@kotlin.internal.InlineOnly\npublic actual inline fun String.toFloatOrNull(): Float? = toDoubleOrNull().unsafeCast<Float?>()\n\n/\*\*\n \* Returns a string representation of this [Byte] value in the specified [radix].\n \* @throws IllegalArgumentException when [radix] is not a valid radix for number to string conversion.\n \*/\n@SinceKotlin("1.2")\n@kotlin.internal.InlineOnly\npublic actual inline fun Byte.toString(radix: Int): String = this.toInt().toString(radix)\n\n/\*\*\n \* Returns a string representation of this [Short] value in the specified [radix].\n \* @throws IllegalArgumentException when [radix] is not a valid radix for number to string conversion.\n \*/\n@SinceKotlin("1.2")\n@kotlin.internal.InlineOnly\npublic actual inline fun Short.toString(radix: Int): String = this.toInt().toString(radix)\n\n/\*\*\n \* Returns a string representation of this [Int] value in the specified [radix].\n \* @throws IllegalArgumentException when [radix] is not a valid radix for number to string conversion.\n \*/\n@SinceKotlin("1.2")\npublic actual fun Int.toString(radix: Int): String = asDynamic().toString(checkRadix(radix))\n\nprivate fun String.isNaN(): Boolean = when (this.lowercase()) {\n "nan", "+nan", "-nan" -> true\n else -> false\n}\n\n/\*\*\n \* Checks whether the given [radix] is valid radix for string to number and number to string conversion.\n \*/\n@PublishedApi\ninternal actual fun checkRadix(radix: Int):

```

Int {\n  if (radix !in 2..36) {\n    throw IllegalArgumentException("\radix $radix was not in valid range 2..36")\n  }\n  return radix\n}\n\ninternal actual fun digitOf(char: Char, radix: Int): Int = when {\n  char >= '0' && char <= '9' -> char - '0'\n  char >= 'A' && char <= 'Z' -> char - 'A' + 10\n  char >= 'a' && char <= 'z' -> char - 'a' + 10\n  char < "\u0080" -> -1\n  char >= "\uFF21" && char <= "\uFF3A" -> char - "\uFF21" + 10 // full-width latin capital letter\n  char >= "\uFF41" && char <= "\uFF5A" -> char - "\uFF41" + 10 // full-width latin small letter\n  else -> char.digitToIntImpl()\n}.let { if (it >= radix) -1 else it }\n", "\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n *\n\npackage kotlin.text\n\nimport kotlin.js.RegExp\n\n/**\n * Provides enumeration values to use to set regular expression options.\n *\npublic actual enum class RegexOptions(val value: String) {\n  /** Enables case-insensitive matching. */\n  IGNORE_CASE("i"),\n  /** Enables multiline mode.\n   *\n   * In multiline mode the expressions `^` and `$` match just after or just before, respectively, a line terminator or the end of the input sequence. */\n  MULTILINE("m")\n}\n\nprivate fun Iterable<RegexOption>.toFlags(prepend: String): String = joinToString("\", prefix = prepend) { it.value }\n\n/**\n * Represents the results from a single capturing group within a [MatchResult] of [Regex].\n *\n @param value The value of captured group.\n *\npublic actual data class MatchGroup(actual val value: String)\n\n/**\n * Represents a compiled regular expression.\n *\n Provides functions to match strings in text with a pattern, replace the found occurrences and split text around matches.\n *\n For pattern syntax reference see [MDN RegExp](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/RegExp#Special_characters_meaning_in_regular_expressions)\n *\n and [http://www.w3schools.com/jsref/jsref_obj_regexp.asp](https://www.w3schools.com/jsref/jsref_obj_regexp.asp).\n *\n Note that `RegExp` objects under the hood are constructed with [the `u` flag](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/RegExp/unicode)\n *\n that enables Unicode-related features in regular expressions. This also makes the pattern syntax more strict,\n *\n for example, prohibiting unnecessary escape sequences.\n *\n @constructor Creates a regular expression from the specified [pattern] string and the specified set of [options].\n *\npublic actual class Regex actual constructor(pattern: String, options: Set<RegexOption>) {\n  /** Creates a regular expression from the specified [pattern] string and the specified single [option]. */\n  public actual constructor(pattern: String, option: RegexOption) : this(pattern, setOf(option))\n  /** Creates a regular expression from the specified [pattern] string and the default options. */\n  public actual constructor(pattern: String) : this(pattern, emptySet())\n  /** The pattern string of this regular expression. */\n  public actual val pattern: String = pattern\n  /** The set of options that were used to create this regular expression. */\n  public actual val options: Set<RegexOption> = options.toSet()\n  private val nativePattern: RegExp = RegExp(pattern, options.toFlags("gu"))\n  private var nativeStickyPattern: RegExp? = null\n  private fun initStickyPattern(): RegExp =\n    nativeStickyPattern ?: RegExp(pattern, options.toFlags("yu"))\n  .also { nativeStickyPattern = it }\n  private var nativeMatchesEntirePattern: RegExp? = null\n  private fun initMatchesEntirePattern(): RegExp =\n    nativeMatchesEntirePattern ?: run {\n      if (pattern.startsWith('^') && pattern.endsWith('$'))\n        nativePattern\n      else\n        return RegExp("^${pattern.trimStart('^').trimEnd('$')}\$"),\n      options.toFlags("gu"))\n    }.also { nativeMatchesEntirePattern = it }\n  /** Indicates whether the regular expression matches the entire [input]. */\n  public actual infix fun matches(input: CharSequence): Boolean {\n    nativePattern.reset()\n    val match = nativePattern.exec(input.toString())\n    return match != null && match.index == 0 && nativePattern.lastIndex == input.length\n  }\n  /** Indicates whether the regular expression can find at least one match in the specified [input]. */\n  public actual fun containsMatchIn(input: CharSequence): Boolean {\n    nativePattern.reset()\n    return nativePattern.test(input.toString())\n  }\n  @SinceKotlin("1.5")\n  @ExperimentalStdlibApi\n  public actual fun matchesAt(input: CharSequence, index: Int): Boolean {\n    if (index < 0 || index > input.length) {\n      throw IndexOutOfBoundsException("index out of bounds: $index, input length: ${input.length}")\n    }\n    val pattern = initStickyPattern()\n    pattern.lastIndex = index\n    return pattern.test(input.toString())\n  }\n  /**\n   *\n Returns the first match of a

```

regular expression in the [input], beginning at the specified [startIndex].

- \* @param startIndex An index to start search with, by default 0. Must be not less than zero and not greater than `input.length()`
- \* @return An instance of [MatchResult] if match was found or `null` otherwise.
- \* @throws IndexOutOfBoundsException if [startIndex] is less than zero or greater than the length of the [input] char sequence.
- \* @sample `samples.text.Regexps.find`

```
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual fun find(input: CharSequence, startIndex: Int = 0): MatchResult? {
    if (startIndex < 0 || startIndex > input.length) {
        throw IndexOutOfBoundsException("Start index out of bounds: $startIndex, input length: ${input.length}")
    }
    return nativePattern.findNext(input.toString(), startIndex, nativePattern)
}
```

- \* Returns a sequence of all occurrences of a regular expression within the [input] string, beginning at the specified [startIndex].
- \* @throws IndexOutOfBoundsException if [startIndex] is less than zero or greater than the length of the [input] char sequence.
- \* @sample `samples.text.Regexps.findAll`

```
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")
public actual fun findAll(input: CharSequence, startIndex: Int = 0): Sequence<MatchResult> {
    if (startIndex < 0 || startIndex > input.length) {
        throw IndexOutOfBoundsException("Start index out of bounds: $startIndex, input length: ${input.length}")
    }
    return generateSequence({ find(input, startIndex) }, { match -> match.next() })
}
```

- \* Attempts to match the entire [input] CharSequence against the pattern.
- \* @return An instance of [MatchResult] if the entire input matches or `null` otherwise.

```
public actual fun matchEntire(input: CharSequence): MatchResult? =
    initMatchesEntirePattern().findNext(input.toString(), 0, nativePattern)
```

@SinceKotlin("1.5")  
@ExperimentalStdlibApi

```
public actual fun matchAt(input: CharSequence, index: Int): MatchResult? {
    if (index < 0 || index > input.length) {
        throw IndexOutOfBoundsException("index out of bounds: $index, input length: ${input.length}")
    }
    return initStickyPattern().findNext(input.toString(), index, nativePattern)
}
```

- \* Replaces all occurrences of this regular expression in the specified [input] string with specified [replacement] expression.
- \* The replacement string may contain references to the captured groups during a match. Occurrences of `$index` in the replacement string will be substituted with the subsequences corresponding to the captured groups with the specified index.
- \* The first digit after '\$' is always treated as part of group reference. Subsequent digits are incorporated into `index` only if they would form a valid group reference. Only the digits '0'..'9' are considered as potential components of the group reference. Note that indexes of captured groups start from 1, and the group with index 0 is the whole match.
- \* Backslash character '\' can be used to include the succeeding character as a literal in the replacement string, e.g. `\\$` or `\\\\`.
- \* [Regex.escapeReplacement] can be used if [replacement] have to be treated as a literal string.
- \* Note that referring named capturing groups by name is currently not supported in Kotlin/JS. However, you can still refer them by index.
- \* @param input the char sequence to find matches of this regular expression in
- \* @param replacement the expression to replace found matches with
- \* @return the result of replacing each occurrence of this regular expression in [input] with the result of evaluating the [replacement] expression
- \* @throws RuntimeException if [replacement] expression is malformed, or capturing group with specified `name` or `index` does not exist

```
public actual fun replace(input: CharSequence, replacement: String): String {
    if (!replacement.contains("\\\\") && !replacement.contains('$')) {
        return input.toString().nativeReplace(nativePattern, replacement)
    }
    return replace(input) { substituteGroupRefs(it, replacement) }
}
```

- \* Replaces all occurrences of this regular expression in the specified [input] string with the result of the given function [transform] that takes [MatchResult] and returns a string to be used as a replacement for that match.

```
public actual fun replace(input: CharSequence, transform: (MatchResult) -> CharSequence): String {
    var match = find(input)
    if (match == null) return input.toString()
    var lastStart = 0
    val length = input.length
    val sb = StringBuilder(length)
    do {
        val foundMatch = match!!
        sb.append(input, lastStart, foundMatch.range.start)
        sb.append(transform(foundMatch))
        lastStart = foundMatch.range.endInclusive + 1
        match = foundMatch.next()
    } while (lastStart < length && match != null)
    if (lastStart < length) {
        sb.append(input, lastStart, length)
    }
    return sb.toString()
}
```

```

sb.toString()\n } \n /*\n * Replaces the first occurrence of this regular expression in the specified [input]
string with specified [replacement] expression.\n *\n * The replacement string may contain references to the
captured groups during a match. Occurrences of `index`\n *\n * in the replacement string will be substituted with the
subsequences corresponding to the captured groups with the specified index.\n *\n * The first digit after '$' is always
treated as part of group reference. Subsequent digits are incorporated\n *\n * into `index` only if they would form a
valid group reference. Only the digits '0'..'9' are considered as potential components\n *\n * of the group reference.
Note that indexes of captured groups start from 1, and the group with index 0 is the whole match.\n *\n *
Backslash character '\\' can be used to include the succeeding character as a literal in the replacement string, e.g, '\\$'
or '\\\\\\'.\n *\n * [Regex.escapeReplacement] can be used if [replacement] have to be treated as a literal string.\n *\n
*\n * Note that referring named capturing groups by name is not supported currently in Kotlin/JS.\n *\n * However, you
can still refer them by index.\n *\n * @param input the char sequence to find a match of this regular expression
in\n *\n * @param replacement the expression to replace the found match with\n *\n * @return the result of replacing
the first occurrence of this regular expression in [input] with the result of evaluating the [replacement] expression\n
*\n * @throws RuntimeException if [replacement] expression is malformed, or capturing group with specified `name`
or `index` does not exist\n *\n
public actual fun replaceFirst(input: CharSequence, replacement: String): String
{\n     if (!replacement.contains("\\\\") && !replacement.contains('$')) {\n         val nonGlobalOptions =
options.toFlags("\\u")\n         return input.toString().nativeReplace(RegExp(pattern, nonGlobalOptions),
replacement)\n     }\n     val match = find(input) ?: return input.toString()\n     return buildString {\n
append(input.substring(0, match.range.first))\n         append(substituteGroupRefs(match, replacement))\n
append(input.substring(match.range.last + 1, input.length))\n     }\n }\n\n /*\n * Splits the [input]
CharSequence to a list of strings around matches of this regular expression.\n *\n * @param limit Non-negative
value specifying the maximum number of substrings the string can be split to.\n *\n * Zero by default means no limit
is set.\n *\n * @Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\n public actual fun
split(input: CharSequence, limit: Int = 0): List<String> {\n     requireNonNegativeLimit(limit)\n     val matches =
findAll(input).let { if (limit == 0) it else it.take(limit - 1) }\n     val result = mutableListOf<String>()\n     var
lastStart = 0\n     for (match in matches) {\n         result.add(input.subSequence(lastStart,
match.range.start).toString())\n         lastStart = match.range.endInclusive + 1\n     }\n     result.add(input.subSequence(lastStart, input.length).toString())\n     return result\n }\n\n /*\n * Splits the
[input] CharSequence to a sequence of strings around matches of this regular expression.\n *\n * @param limit
Non-negative value specifying the maximum number of substrings the string can be split to.\n *\n * Zero by default
means no limit is set.\n *\n * @sample samples.text.Regexps.splitToSequence\n *\n * @SinceKotlin("1.6")\n
@WasExperimental(ExperimentalStdlibApi::class)\n
@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\n public actual fun
splitToSequence(input: CharSequence, limit: Int = 0): Sequence<String> {\n
requireNonNegativeLimit(limit)\n     return sequence {\n         var match = find(input)\n         if (match ==
null || limit == 1) {\n             yield(input.toString())\n             return@sequence\n         }\n         var
nextStart = 0\n         var splitCount = 0\n         do {\n             val foundMatch = match!!\n
yield(input.substring(nextStart, foundMatch.range.first))\n             nextStart = foundMatch.range.endInclusive +
1\n             match = foundMatch.next()\n         } while (++splitCount != limit - 1 && match != null)\n
yield(input.substring(nextStart, input.length))\n     }\n }\n\n /*\n * Returns the string representation of
this regular expression, namely the [pattern] of this regular expression.\n *\n * Note that another regular
expression constructed from the same pattern string may have different [options]\n *\n * and may match strings
differently.\n *\n public override fun toString(): String = nativePattern.toString()\n actual companion object
{\n     /*\n     * Returns a regular expression that matches the specified [literal] string literally.\n     * No
characters of that string will have special meaning when searching for an occurrence of the regular expression.\n
*\n     *\n     public actual fun fromLiteral(literal: String): Regex = Regex(escape(literal))\n     /*\n     * Returns a
regular expression pattern string that matches the specified [literal] string literally.\n     * No characters of that
string will have special meaning when searching for an occurrence of the regular expression.\n     *\n     public

```



```

@ExperimentalStdlibApi\n  constructor(content: String)\n\n  override val length: Int\n\n  override operator fun
get(index: Int): Char\n\n  override fun subSequence(startIndex: Int, endIndex: Int): CharSequence\n\n  override
fun append(value: Char): StringBuilder\n  override fun append(value: CharSequence?): StringBuilder\n  override
fun append(value: CharSequence?, startIndex: Int, endIndex: Int): StringBuilder\n\n  /**\n   * Reverses the
contents of this string builder and returns this instance.\n   *\n   * Surrogate pairs included in this string builder
are treated as single characters.\n   * Therefore, the order of the high-low surrogates is never reversed.\n   *\n   *
Note that the reverse operation may produce new surrogate pairs that were unpaired low-surrogates and high-
surrogates before the operation.\n   * For example, reversing `"\uDC00\uD800"` produces `"\uD800\uDC00"`
which is a valid surrogate pair.\n   */\n  fun reverse(): StringBuilder\n\n  /**\n   * Appends the string
representation of the specified object [value] to this string builder and returns this instance.\n   *\n   * The overall
effect is exactly as if the [value] were converted to a string by the `value.toString()` method,\n   * and then that
string was appended to this string builder.\n   */\n  fun append(value: Any?): StringBuilder\n\n  /**\n   *
Appends the string representation of the specified boolean [value] to this string builder and returns this instance.\n
   *\n   * The overall effect is exactly as if the [value] were converted to a string by the `value.toString()` method,\n
   * and then that string was appended to this string builder.\n   */\n  @SinceKotlin("1.3")\n  fun append(value:
Boolean): StringBuilder\n\n  /**\n   * Appends characters in the specified character array [value] to this string
builder and returns this instance.\n   *\n   * Characters are appended in order, starting at the index 0.\n   */\n
  @SinceKotlin("1.4")\n  @WasExperimental(ExperimentalStdlibApi::class)\n  fun append(value: CharArray):
StringBuilder\n\n  /**\n   * Appends the specified string [value] to this string builder and returns this instance.\n
   *\n   * If [value] is `null`, then the four characters `"\u0000\u0000\u0000\u0000"` are appended.\n   */\n  @SinceKotlin("1.3")\n  fun
append(value: String?): StringBuilder\n\n  /**\n   * Returns the current capacity of this string builder.\n   *\n   *
The capacity is the maximum length this string builder can have before an allocation occurs.\n   */\n  @SinceKotlin("1.3")\n  // @ExperimentalStdlibApi\n  @Deprecated("Obtaining StringBuilder capacity is not
supported in JS and common code.", level = DeprecationLevel.ERROR)\n  fun capacity(): Int\n\n  /**\n   *
Ensures that the capacity of this string builder is at least equal to the specified [minimumCapacity].\n   *\n   * If
the current capacity is less than the [minimumCapacity], a new backing storage is allocated with greater capacity.\n
   * Otherwise, this method takes no action and simply returns.\n   */\n  @SinceKotlin("1.4")\n  @WasExperimental(ExperimentalStdlibApi::class)\n  fun ensureCapacity(minimumCapacity: Int)\n\n  /**\n   *
Returns the index within this string builder of the first occurrence of the specified [string].\n   *\n   * Returns `-1`
if the specified [string] does not occur in this string builder.\n   */\n  @SinceKotlin("1.4")\n  @WasExperimental(ExperimentalStdlibApi::class)\n  fun indexOf(string: String): Int\n\n  /**\n   * Returns the
index within this string builder of the first occurrence of the specified [string],\n   * starting at the specified
[startIndex].\n   *\n   * Returns `-1` if the specified [string] does not occur in this string builder starting at the
specified [startIndex].\n   */\n  @SinceKotlin("1.4")\n  @WasExperimental(ExperimentalStdlibApi::class)\n  fun
indexOf(string: String, startIndex: Int): Int\n\n  /**\n   * Returns the index within this string builder of the last
occurrence of the specified [string].\n   * The last occurrence of empty string `""` is considered to be at the index
equal to `this.length`.\n   *\n   * Returns `-1` if the specified [string] does not occur in this string builder.\n
   */\n  @SinceKotlin("1.4")\n  @WasExperimental(ExperimentalStdlibApi::class)\n  fun lastIndexOf(string: String):
Int\n\n  /**\n   * Returns the index within this string builder of the last occurrence of the specified [string],\n   *
starting from the specified [startIndex] toward the beginning.\n   *\n   * Returns `-1` if the specified [string] does
not occur in this string builder starting at the specified [startIndex].\n   */\n  @SinceKotlin("1.4")\n  @WasExperimental(ExperimentalStdlibApi::class)\n  fun lastIndexOf(string: String, startIndex: Int): Int\n\n  /**\n   * Inserts the string representation of the specified boolean [value] into this string builder at the specified
[index] and returns this instance.\n   *\n   * The overall effect is exactly as if the [value] were converted to a string
by the `value.toString()` method,\n   * and then that string was inserted into this string builder at the specified
[index].\n   *\n   * @throws IndexOutOfBoundsException if [index] is less than zero or greater than the length of
this string builder.\n   */\n  @SinceKotlin("1.4")\n  @WasExperimental(ExperimentalStdlibApi::class)\n  fun
insert(index: Int, value: Boolean): StringBuilder\n\n  /**\n   * Inserts the specified character [value] into this

```

```

string builder at the specified [index] and returns this instance.\n * \n * @throws IndexOutOfBoundsException
if [index] is less than zero or greater than the length of this string builder.\n * \n @SinceKotlin("1.4")\n
@WasExperimental(ExperimentalStdlibApi::class)\n fun insert(index: Int, value: Char): StringBuidler\n\n /**\n
* Inserts characters in the specified character array [value] into this string builder at the specified [index] and
returns this instance.\n * \n * The inserted characters go in same order as in the [value] character array, starting
at [index].\n * \n * @throws IndexOutOfBoundsException if [index] is less than zero or greater than the length
of this string builder.\n * \n @SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n
fun insert(index: Int, value: CharArray): StringBuidler\n\n /**\n * Inserts characters in the specified character
sequence [value] into this string builder at the specified [index] and returns this instance.\n * \n * The inserted
characters go in the same order as in the [value] character sequence, starting at [index].\n * \n * @param index
the position in this string builder to insert at.\n * @param value the character sequence from which characters are
inserted. If [value] is `null`, then the four characters `"\u0000\u0000\u0000\u0000"` are inserted.\n * \n * @throws
IndexOutOfBoundsException if [index] is less than zero or greater than the length of this string builder.\n * \n
@SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n fun insert(index: Int, value:
CharSequence?): StringBuidler\n\n /**\n * Inserts the string representation of the specified object [value] into
this string builder at the specified [index] and returns this instance.\n * \n * The overall effect is exactly as if the
[value] were converted to a string by the `value.toString()` method,\n * and then that string was inserted into this
string builder at the specified [index].\n * \n * @throws IndexOutOfBoundsException if [index] is less than
zero or greater than the length of this string builder.\n * \n @SinceKotlin("1.4")\n
@WasExperimental(ExperimentalStdlibApi::class)\n fun insert(index: Int, value: Any?): StringBuidler\n\n /**\n
* Inserts the string [value] into this string builder at the specified [index] and returns this instance.\n * \n * If
[value] is `null`, then the four characters `"\u0000\u0000\u0000\u0000"` are inserted.\n * \n * @throws IndexOutOfBoundsException
if [index] is less than zero or greater than the length of this string builder.\n * \n @SinceKotlin("1.4")\n
@WasExperimental(ExperimentalStdlibApi::class)\n fun insert(index: Int, value: String?): StringBuidler\n\n
/**\n * Sets the length of this string builder to the specified [newLength].\n * \n * If the [newLength] is less
than the current length, it is changed to the specified [newLength].\n * \n * Otherwise, null characters `"\u0000"` are
appended to this string builder until its length is less than the [newLength].\n * \n * Note that in Kotlin/JS [set]
operator function has non-constant execution time complexity.\n * \n * Therefore, increasing length of this string
builder and then updating each character by index may slow down your program.\n * \n * @throws
IndexOutOfBoundsException or [IllegalArgumentException] if [newLength] is less than zero.\n * \n
@SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n fun setLength(newLength:
Int)\n\n /**\n * Returns a new [String] that contains characters in this string builder at [startIndex] (inclusive)
and up to the [length] (exclusive).\n * \n * @throws IndexOutOfBoundsException if [startIndex] is less than
zero or greater than the length of this string builder.\n * \n @SinceKotlin("1.4")\n
@WasExperimental(ExperimentalStdlibApi::class)\n fun substring(startIndex: Int): String\n\n /**\n * Returns
a new [String] that contains characters in this string builder at [startIndex] (inclusive) and up to the [endIndex]
(exclusive).\n * \n * @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex]
or [endIndex] is out of range of this string builder indices or when `startIndex > endIndex`.\n * \n
@SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n fun substring(startIndex: Int,
endIndex: Int): String\n\n /**\n * Attempts to reduce storage used for this string builder.\n * \n * If the
backing storage of this string builder is larger than necessary to hold its current contents,\n * then it may be
resized to become more space efficient.\n * \n * Calling this method may, but is not required to, affect the value of the
[capacity] property.\n * \n @SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\n
fun trimToSize()\n\n /**\n * Clears the content of this string builder making it empty and returns this instance.\n\n
*\n * @sample samples.text.Strings.clearStringBuilder\n * \n @SinceKotlin("1.3")\n\n public expect fun
StringBuilder.clear(): StringBuidler\n\n /**\n * Sets the character at the specified [index] to the specified [value].\n\n
*\n * @throws IndexOutOfBoundsException if [index] is out of bounds of this string builder.\n\n
*\n * \n @SinceKotlin("1.4")\n\n @WasExperimental(ExperimentalStdlibApi::class)\n\n public expect operator fun

```

`StringBuilder.set(index: Int, value: Char)`\n\n\*\*\n \* Replaces characters in the specified range of this string builder with characters in the specified string [value] and returns this instance.\n \*\n \* @param startIndex the beginning (inclusive) of the range to replace.\n \* @param endIndex the end (exclusive) of the range to replace.\n \* @param value the string to replace with.\n \*\n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] if [startIndex] is less than zero, greater than the length of this string builder, or `startIndex > endIndex`.\n

`*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun`

`StringBuilder.setRange(startIndex: Int, endIndex: Int, value: String): StringBuilder`\n\n\*\*\n \* Removes the character at the specified [index] from this string builder and returns this instance.\n \*\n \* If the `Char` at the specified [index] is part of a supplementary code point, this method does not remove the entire supplementary character.\n \*\n \* @param index the index of `Char` to remove.\n \*\n \* @throws IndexOutOfBoundsException if [index] is out of bounds of this string builder.\n

`*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun`

`StringBuilder.deleteAt(index: Int): StringBuilder`\n\n\*\*\n \* Removes characters in the specified range from this string builder and returns this instance.\n \*\n \* @param startIndex the beginning (inclusive) of the range to remove.\n \* @param endIndex the end (exclusive) of the range to remove.\n \*\n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] is out of range of this string builder indices or when `startIndex > endIndex`.\n

`*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun`

`StringBuilder.deleteRange(startIndex: Int, endIndex: Int): StringBuilder`\n\n\*\*\n \* Copies characters from this string builder into the [destination] character array.\n \*\n \* @param destination the array to copy to.\n \* @param destinationOffset the position in the array to copy to, 0 by default.\n \* @param startIndex the beginning (inclusive) of the range to copy, 0 by default.\n \* @param endIndex the end (exclusive) of the range to copy, length of this string builder by default.\n \*\n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of this string builder indices or when `startIndex > endIndex`.\n \* @throws IndexOutOfBoundsException when the subrange doesn't fit into the [destination] array starting at the specified [destinationOffset],\n \* or when that index is out of the [destination] array indices range.\n

`*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun`

`StringBuilder.toCharArray(destination: CharArray, destinationOffset: Int = 0, startIndex: Int = 0, endIndex: Int = this.length)`\n\n\*\*\n \* Appends characters in a subarray of the specified character array [value] to this string builder and returns this instance.\n \*\n \* Characters are appended in order, starting at specified [startIndex].\n \*\n \* @param value the array from which characters are appended.\n \* @param startIndex the beginning (inclusive) of the subarray to append.\n \* @param endIndex the end (exclusive) of the subarray to append.\n \*\n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of the [value] array indices or when `startIndex > endIndex`.\n

`*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun`

`StringBuilder.appendRange(value: CharArray, startIndex: Int, endIndex: Int): StringBuilder`\n\n\*\*\n \* Appends a subsequence of the specified character sequence [value] to this string builder and returns this instance.\n \*\n \* @param value the character sequence from which a subsequence is appended.\n \* @param startIndex the beginning (inclusive) of the subsequence to append.\n \* @param endIndex the end (exclusive) of the subsequence to append.\n \*\n \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of the [value] character sequence indices or when `startIndex > endIndex`.\n

`*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun`

`StringBuilder.appendRange(value: CharSequence, startIndex: Int, endIndex: Int): StringBuilder`\n\n\*\*\n \* Inserts characters in a subarray of the specified character array [value] into this string builder at the specified [index] and returns this instance.\n \*\n \* The inserted characters go in same order as in the [value] array, starting at [index].\n \*\n \* @param index the position in this string builder to insert at.\n \* @param value the array from which characters are inserted.\n \* @param startIndex the beginning (inclusive) of the subarray to insert.\n \* @param endIndex the end (exclusive) of the subarray to insert.\n \*\n \* @throws IndexOutOfBoundsException or

[IllegalArgumentException] when [startIndex] or [endIndex] is out of range of the [value] array indices or when `startIndex > endIndex`. \n \* @throws IndexOutOfBoundsException if [index] is less than zero or greater than the length of this string builder. \n

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun  
StringBuilder.insertRange(index: Int, value: CharArray, startIndex: Int, endIndex: Int): StringBuilder\n\n/**\n * Inserts characters in a subsequence of the specified character sequence [value] into this string builder at the specified [index] and returns this instance. \n * \n * The inserted characters go in the same order as in the [value] character sequence, starting at [index]. \n * \n * @param index the position in this string builder to insert at. \n * @param value the character sequence from which a subsequence is inserted. \n * @param startIndex the beginning (inclusive) of the subsequence to insert. \n * @param endIndex the end (exclusive) of the subsequence to insert. \n * \n * @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of the [value] character sequence indices or when `startIndex > endIndex`. \n * @throws IndexOutOfBoundsException if [index] is less than zero or greater than the length of this string builder. \n
```

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun  
StringBuilder.insertRange(index: Int, value: CharSequence, startIndex: Int, endIndex: Int):  
StringBuilder\n\n@Suppress("EXTENSION_SHADOWED_BY_MEMBER")\n@Deprecated("Use  
append(value: Any?) instead", ReplaceWith("append(value = obj)"),  
DeprecationLevel.WARNING)\n@kotlin.internal.InlineOnly\npublic inline fun StringBuilder.append(obj: Any?):  
StringBuilder = this.append(obj)\n\n/**\n * Builds new string by populating newly created [StringBuilder] using  
provided [builderAction] \n * and then converting it to [String]. \n */\n@kotlin.internal.InlineOnly\npublic inline fun  
buildString(builderAction: StringBuilder.() -> Unit): String {\n    contract { callsInPlace(builderAction,  
InvocationKind.EXACTLY_ONCE) }\n    return StringBuilder().apply(builderAction).toString()\n}\n\n/**\n * Builds new string by populating newly created [StringBuilder] initialized with the given [capacity] \n * using  
provided [builderAction] and then converting it to [String]. \n
```

```
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline fun buildString(capacity: Int, builderAction:  
StringBuilder.() -> Unit): String {\n    contract { callsInPlace(builderAction, InvocationKind.EXACTLY_ONCE)  
}\n    return StringBuilder(capacity).apply(builderAction).toString()\n}\n\n/**\n * Appends all arguments to the  
given StringBuilder. \n */\npublic fun StringBuilder.append(vararg value: String?): StringBuilder {\n    for (item in  
value)\n        append(item)\n    return this\n}\n\n/**\n * Appends all arguments to the given StringBuilder. \n */\npublic fun StringBuilder.append(vararg value: Any?): StringBuilder {\n    for (item in value)\n        append(item)\n    return this\n}\n\n/**\n * Appends a line feed character (`\n`) to this StringBuilder. \n
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun StringBuilder.appendLine():  
StringBuilder = append("\n")\n\n/**\n * Appends [value] to this [StringBuilder], followed by a line feed character  
(`n`). \n */\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun StringBuilder.appendLine(value:  
CharSequence?): StringBuilder = append(value).appendLine()\n\n/**\n * Appends [value] to this [StringBuilder],  
followed by a line feed character (`n`). \n */\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun  
StringBuilder.appendLine(value: String?): StringBuilder = append(value).appendLine()\n\n/**\n * Appends [value] to  
this [StringBuilder], followed by a line feed character (`n`). \n
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun StringBuilder.appendLine(value: Any?):  
StringBuilder = append(value).appendLine()\n\n/**\n * Appends [value] to this [StringBuilder], followed by a line feed  
character (`n`). \n */\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun  
StringBuilder.appendLine(value: CharArray): StringBuilder = append(value).appendLine()\n\n/**\n * Appends [value]  
to this [StringBuilder], followed by a line feed character (`n`). \n
```

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun StringBuilder.appendLine(value: Char):  
StringBuilder = append(value).appendLine()\n\n/**\n * Appends [value] to this [StringBuilder], followed by a line feed  
character (`n`). \n */\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun
```

```
StringBuilder.appendLine(value: Boolean): StringBuilder = append(value).appendLine()\n\n"/>\n * Copyright 2010-  
2021 JetBrains s.r.o. and Kotlin Programming Language contributors. \n * Use of this source code is governed by the
```

```

Apache 2.0 license that can be found in the license/LICENSE.txt file.\n *\n\npackage kotlin.text\n\nimport
kotlin.js.RegExp\n\n@kotlin.internal.InlineOnly\n\ninternal actual inline fun String.nativeIndexOf(ch: Char,
fromIndex: Int): Int = nativeIndexOf(ch.toString(), fromIndex)\n\n@kotlin.internal.InlineOnly\n\ninternal actual
inline fun String.nativeLastIndexOf(ch: Char, fromIndex: Int): Int = nativeLastIndexOf(ch.toString(),
fromIndex)\n\n/**\n * Returns `true` if this string starts with the specified prefix.\n
*\n\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\n\npublic actual fun
String.startsWith(prefix: String, ignoreCase: Boolean = false): Boolean {\n    if (!ignoreCase)\n        return
nativeStartsWith(prefix, 0)\n    else\n        return regionMatches(0, prefix, 0, prefix.length, ignoreCase)\n}\n\n/**\n *
Returns `true` if a substring of this string starting at the specified offset [startIndex] starts with the specified prefix.\n
*\n\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\n\npublic actual fun
String.startsWith(prefix: String, startIndex: Int, ignoreCase: Boolean = false): Boolean {\n    if (!ignoreCase)\n
return nativeStartsWith(prefix, startIndex)\n    else\n        return regionMatches(startIndex, prefix, 0, prefix.length,
ignoreCase)\n}\n\n/**\n * Returns `true` if this string ends with the specified suffix.\n
*\n\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\n\npublic actual fun
String.endsWith(suffix: String, ignoreCase: Boolean = false): Boolean {\n    if (!ignoreCase)\n        return
nativeEndsWith(suffix)\n    else\n        return regionMatches(length - suffix.length, suffix, 0, suffix.length,
ignoreCase)\n}\n\n@Deprecated("Use Regex.matches() instead",
ReplaceWith("regex.toRegex().matches(this)"))\n\n@DeprecatedSinceKotlin(warningSince = "1.6")\n\npublic fun
String.matches(regex: String): Boolean {\n    @Suppress("DEPRECATION")\n    val result = this.match(regex)\n
return result != null && result.size != 0\n}\n\n/**\n * Returns `true` if this string is empty or consists solely of
whitespace characters.\n *\n * @sample samples.text.Strings.stringIsBlank\n *\n\npublic actual fun
CharSequence.isBlank(): Boolean = length == 0 || indices.all { this[it].isWhitespace() }\n\n/**\n * Returns `true` if
this string is equal to [other], optionally ignoring character case.\n *\n * Two strings are considered to be equal if
they have the same length and the same character at the same index.\n *\n * If [ignoreCase] is true, the result of
`Char.toUpperCaseChar().toLowerCaseChar()` on each character is compared.\n *\n * @param ignoreCase `true` to ignore
character case when comparing strings. By default `false`.\n
*\n\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\n\npublic actual fun
String?.equals(other: String?, ignoreCase: Boolean = false): Boolean {\n    if (this == null) return other == null\n    if
(other == null) return false\n    if (!ignoreCase) return this == other\n    if (this.length != other.length) return
false\n\n    for (index in 0 until this.length) {\n        val thisChar = this[index]\n        val otherChar = other[index]\n
if (!thisChar.equals(otherChar, ignoreCase)) {\n            return false\n        }\n    }\n\n    return
true\n}\n\n@Suppress("ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS")\n\npublic actual fun
CharSequence.regionMatches(thisOffset: Int, other: CharSequence, otherOffset: Int, length: Int, ignoreCase:
Boolean = false): Boolean =\n    regionMatchesImpl(thisOffset, other, otherOffset, length, ignoreCase)\n\n/**\n *
Returns a copy of this string having its first letter titlecased using the rules of the default locale,\n *\n * or the original
string if it's empty or already starts with a title case letter.\n *\n * The title case of a character is usually the same as
its upper case with several exceptions.\n *\n * The particular list of characters with the special title case form depends
on the underlying platform.\n *\n * @sample samples.text.Strings.capitalize\n *\n\n@Deprecated("Use
replaceFirstChar instead.", ReplaceWith("replaceFirstChar { if (it.isLowerCase()) it.titlecase() else it.toString()
}"))\n\n@DeprecatedSinceKotlin(warningSince = "1.5")\n\npublic actual fun String.capitalize(): String {\n    return if
(isNotEmpty()) substring(0, 1).uppercase() + substring(1) else this\n}\n\n/**\n * Returns a copy of this string having
its first letter lowercased using the rules of the default locale,\n *\n * or the original string if it's empty or already
starts with a lower case letter.\n *\n * @sample samples.text.Strings.decitalize\n *\n\n@Deprecated("Use
replaceFirstChar instead.", ReplaceWith("replaceFirstChar { it.lowercase()
}"))\n\n@DeprecatedSinceKotlin(warningSince = "1.5")\n\npublic actual fun String.decitalize(): String {\n    return
if (isNotEmpty()) substring(0, 1).lowercase() + substring(1) else this\n}\n\n/**\n * Returns a string containing this
char sequence repeated [n] times.\n *\n * @throws [IllegalArgumentException] when n < 0.\n *\n * @sample
samples.text.Strings.repeat\n *\n\npublic actual fun CharSequence.repeat(n: Int): String {\n    require(n >= 0) {

```

```

\Count 'n' must be non-negative, but was $n.\} \return when (n) {n 0 -> \"\n 1 -> this.toString()\n
else -> {\n var result = \"\n if (!isEmpty()) {\n var s = this.toString()\n var count =
n\n while (true) {\n if ((count and 1) == 1) {\n result += s\n }\n
count = count ushr 1\n if (count == 0) {\n break\n }\n s +=
s\n }\n }\n return result\n }\n }\n\n/**\n * Returns a new string obtained by
replacing all occurrences of the [oldValue] substring in this string\n * with the specified [newValue] string.\n *\n *
@sample samples.text.Strings.replace\n
*/\n@Suppress(\"ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS\")\npublic actual fun
String.replace(oldValue: String, newValue: String, ignoreCase: Boolean = false): String =\n
nativeReplace(RegExp(Regex.escape(oldValue), if (ignoreCase) \"gui\" else \"gu\"),
Regex.nativeEscapeReplacement(newValue))\n\n/**\n * Returns a new string with all occurrences of [oldChar]
replaced with [newChar].\n *\n * @sample samples.text.Strings.replace\n
*/\n@Suppress(\"ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS\")\npublic actual fun
String.replace(oldChar: Char, newChar: Char, ignoreCase: Boolean = false): String =\n
nativeReplace(RegExp(Regex.escape(oldChar.toString()), if (ignoreCase) \"gui\" else \"gu\"),
newChar.toString())\n\n@Suppress(\"ACTUAL_FUNCTION_WITH_DEFAULT_ARGUMENTS\")\npublic actual
fun String.replaceFirst(oldValue: String, newValue: String, ignoreCase: Boolean = false): String =\n
nativeReplace(RegExp(Regex.escape(oldValue), if (ignoreCase) \"ui\" else \"u\"),
Regex.nativeEscapeReplacement(newValue))\n\n@Suppress(\"ACTUAL_FUNCTION_WITH_DEFAULT_ARGU
MENTS\")\npublic actual fun String.replaceFirst(oldChar: Char, newChar: Char, ignoreCase: Boolean = false):
String =\n nativeReplace(RegExp(Regex.escape(oldChar.toString()), if (ignoreCase) \"ui\" else \"u\"),
newChar.toString())\n\n\n/**\n * Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\npackage kotlin.text\n\n/**\n * Returns the negative [size] if [throwOnMalformed] is
false, throws [CharacterCodingException] otherwise. *\nprivate fun malformed(size: Int, index: Int,
throwOnMalformed: Boolean): Int {\n if (throwOnMalformed) throw CharacterCodingException(\"Malformed
sequence starting at ${index - 1}\")\n return -size\n}\n\n/**\n * Returns code point corresponding to UTF-16
surrogate pair,\n * where the first of the pair is the [high] and the second is in the [string] at the [index].\n * Returns
zero if the pair is malformed and [throwOnMalformed] is false.\n *\n * @throws CharacterCodingException if the
pair is malformed and [throwOnMalformed] is true.\n *\nprivate fun codePointFromSurrogate(string: String, high:
Int, index: Int, endIndex: Int, throwOnMalformed: Boolean): Int {\n if (high !in 0xD800..0xDBFF || index >=
endIndex) {\n return malformed(0, index, throwOnMalformed)\n }\n val low = string[index].code\n if
(low !in 0xDC00..0xDFFF) {\n return malformed(0, index, throwOnMalformed)\n }\n return 0x10000 +
((high and 0x3FF) shl 10) or (low and 0x3FF)\n}\n\n/**\n * Returns code point corresponding to UTF-8 sequence of
two bytes,\n * where the first byte of the sequence is the [byte1] and the second byte is in the [bytes] array at the
[index].\n * Returns zero if the sequence is malformed and [throwOnMalformed] is false.\n *\n * @throws
CharacterCodingException if the sequence of two bytes is malformed and [throwOnMalformed] is true.\n
*\nprivate fun codePointFrom2(bytes: ByteArray, byte1: Int, index: Int, endIndex: Int, throwOnMalformed:
Boolean): Int {\n if (byte1 and 0x1E == 0 || index >= endIndex) {\n return malformed(0, index,
throwOnMalformed)\n }\n val byte2 = bytes[index].toInt()\n if (byte2 and 0xC0 != 0x80) {\n return
malformed(0, index, throwOnMalformed)\n }\n return (byte1 shl 6) xor byte2 xor 0xF80\n}\n\n/**\n * Returns
code point corresponding to UTF-8 sequence of three bytes,\n * where the first byte of the sequence is the [byte1]
and the others are in the [bytes] array starting from the [index].\n * Returns a non-positive value indicating number
of bytes from [bytes] included in malformed sequence\n * if the sequence is malformed and [throwOnMalformed] is
false.\n *\n * @throws CharacterCodingException if the sequence of three bytes is malformed and
[throwOnMalformed] is true.\n *\nprivate fun codePointFrom3(bytes: ByteArray, byte1: Int, index: Int, endIndex:
Int, throwOnMalformed: Boolean): Int {\n if (index >= endIndex) {\n return malformed(0, index,
throwOnMalformed)\n }\n val byte2 = bytes[index].toInt()\n if (byte1 and 0xF == 0) {\n if (byte2 and

```

```

0xE0 != 0xA0) {\n        // Non-shortest form\n        return malformed(0, index, throwOnMalformed)\n    }\n    } else if (byte1 and 0xF == 0xD) {\n        if (byte2 and 0xE0 != 0x80) {\n            // Surrogate code point\n            return malformed(0, index, throwOnMalformed)\n        }\n    } else if (byte2 and 0xC0 != 0x80) {\n        return\n        malformed(0, index, throwOnMalformed)\n    }\n    }\n    if (index + 1 == endIndex) {\n        return malformed(1, index,\n        throwOnMalformed)\n    }\n    val byte3 = bytes[index + 1].toInt()\n    if (byte3 and 0xC0 != 0x80) {\n        return\n        malformed(1, index, throwOnMalformed)\n    }\n    return (byte1 shl 12) xor (byte2 shl 6) xor byte3 xor -\n    0x1E080)\n}\n\n/**\n * Returns code point corresponding to UTF-8 sequence of four bytes,\n * where the first byte\n * of the sequence is the [byte1] and the others are in the [bytes] array starting from the [index].\n * Returns a non-\n * positive value indicating number of bytes from [bytes] included in malformed sequence\n * if the sequence is\n * malformed and [throwOnMalformed] is false.\n * @throws CharacterCodingException if the sequence of four\n * bytes is malformed and [throwOnMalformed] is true.\n */\nprivate fun codePointFrom4(bytes: ByteArray, byte1:\n    Int, index: Int, endIndex: Int, throwOnMalformed: Boolean): Int {\n    if (index >= endIndex) {\n        malformed(0,\n        index, throwOnMalformed)\n    }\n    val byte2 = bytes[index].toInt()\n    if (byte1 and 0xF == 0x0) {\n        if\n        (byte2 and 0xF0 <= 0x80) {\n            // Non-shortest form\n            return malformed(0, index,\n            throwOnMalformed)\n        }\n    } else if (byte1 and 0xF == 0x4) {\n        if (byte2 and 0xF0 != 0x80) {\n            //\n            Out of Unicode code points domain (larger than U+10FFFF)\n            return malformed(0, index,\n            throwOnMalformed)\n        }\n    } else if (byte1 and 0xF > 0x4) {\n        return malformed(0, index,\n        throwOnMalformed)\n    } else if (byte2 and 0xC0 != 0x80) {\n        return malformed(0, index,\n        throwOnMalformed)\n    }\n    }\n    if (index + 1 == endIndex) {\n        return malformed(1, index,\n        throwOnMalformed)\n    }\n    val byte3 = bytes[index + 1].toInt()\n    if (byte3 and 0xC0 != 0x80) {\n        return\n        malformed(1, index, throwOnMalformed)\n    }\n    }\n    if (index + 2 == endIndex) {\n        return malformed(2, index,\n        throwOnMalformed)\n    }\n    val byte4 = bytes[index + 2].toInt()\n    if (byte4 and 0xC0 != 0x80) {\n        return\n        malformed(2, index, throwOnMalformed)\n    }\n    return (byte1 shl 18) xor (byte2 shl 12) xor (byte3 shl 6) xor\n    byte4 xor 0x381F80)\n}\n\n/**\n * Maximum number of bytes needed to encode a single char.\n * Code points in\n * `0..0x7F` are encoded in a single byte.\n * Code points in `0x80..0x7FF` are encoded in two bytes.\n * Code points\n * in `0x800..0xD7FF` or in `0xE000..0xFFFF` are encoded in three bytes.\n * Surrogate code points in\n * `0xD800..0xDFFF` are not Unicode scalar values, therefore aren't encoded.\n * Code points in\n * `0x10000..0x10FFFF` are represented by a pair of surrogate `Char`s and are encoded in four bytes.\n */\nprivate\n    const val MAX_BYTES_PER_CHAR = 3\n\n/**\n * The byte sequence a malformed UTF-16 char sequence is\n * replaced by.\n */\nprivate val REPLACEMENT_BYTE_SEQUENCE: ByteArray = byteArrayOf(0xEF.toByte(),\n    0xBF.toByte(), 0xBD.toByte())\n\n/**\n * Encodes the [string] using UTF-8 and returns the resulting [ByteArray].\n * @param string the string to encode.\n * @param startIndex the start offset (inclusive) of the substring to\n * encode.\n * @param endIndex the end offset (exclusive) of the substring to encode.\n * @param\n * throwOnMalformed whether to throw on malformed char sequence or replace by the\n * [REPLACEMENT_BYTE_SEQUENCE].\n * @throws CharacterCodingException if the char sequence is\n * malformed and [throwOnMalformed] is true.\n */\ninternal fun encodeUtf8(string: String, startIndex: Int, endIndex:\n    Int, throwOnMalformed: Boolean): ByteArray {\n    require(startIndex >= 0 && endIndex <= string.length &&\n    startIndex <= endIndex)\n    val bytes = ByteArray((endIndex - startIndex) * MAX_BYTES_PER_CHAR)\n    var\n    byteIndex = 0\n    var charIndex = startIndex\n    while (charIndex < endIndex) {\n        val code =\n        string[charIndex++].code\n        when {\n            code < 0x80 -> bytes[byteIndex++] = code.toByte()\n            code < 0x800 -> {\n                bytes[byteIndex++] = ((code shr 6) or 0xC0).toByte()\n                bytes[byteIndex++] = ((code and 0x3F) or 0x80).toByte()\n            }\n            code < 0xD800 || code >= 0xE000 ->\n            {\n                bytes[byteIndex++] = ((code shr 12) or 0xE0).toByte()\n                bytes[byteIndex++] = (((code shr 6)\n                and 0x3F) or 0x80).toByte()\n                bytes[byteIndex++] = ((code and 0x3F) or 0x80).toByte()\n            }\n            else -> { // Surrogate char value\n                val codePoint = codePointFromSurrogate(string, code, charIndex,\n                endIndex, throwOnMalformed)\n                if (codePoint <= 0) {\n                    bytes[byteIndex++] =\n                    REPLACEMENT_BYTE_SEQUENCE[0]\n                    bytes[byteIndex++] =\n                    REPLACEMENT_BYTE_SEQUENCE[1]\n                }\n            }\n        }\n    }\n}

```

```

REPLACEMENT_BYTE_SEQUENCE[2]\n          } else {\n          bytes[byteIndex++] = ((codePoint shr
18) or 0xF0).toByte()\n          bytes[byteIndex++] = (((codePoint shr 12) and 0x3F) or 0x80).toByte()\n          bytes[byteIndex++] = (((codePoint shr 6) and 0x3F) or 0x80).toByte()\n          bytes[byteIndex++] =
(((codePoint and 0x3F) or 0x80).toByte()\n          charIndex++\n          }\n          }\n          }\n          }\n
return if (bytes.size == byteIndex) bytes else bytes.copyOf(byteIndex)\n}\n\n/**\n * The character a malformed
UTF-8 byte sequence is replaced by.\n */\nprivate const val REPLACEMENT_CHAR = "\uFFFF"\n\n/**\n *
Decodes the UTF-8 [bytes] array and returns the resulting [String].\n */\n * @param bytes the byte array to decode.\n
* @param startIndex the start offset (inclusive) of the array to be decoded.\n * @param endIndex the end offset
(exclusive) of the array to be encoded.\n * @param throwOnMalformed whether to throw on malformed byte
sequence or replace by the [REPLACEMENT_CHAR].\n */\n * @throws CharacterCodingException if the array is
malformed UTF-8 byte sequence and [throwOnMalformed] is true.\n */\ninternal fun decodeUtf8(bytes: ByteArray,
startIndex: Int, endIndex: Int, throwOnMalformed: Boolean): String {\n    require(startIndex >= 0 && endIndex <=
bytes.size && startIndex <= endIndex)\n    var byteIndex = startIndex\n    val stringBuilder = StringBuilder()\n
while (byteIndex < endIndex) {\n        val byte = bytes[byteIndex++].toInt()\n        when {\n            byte >= 0 ->\n
stringBuilder.append(byte.toChar())\n            byte shr 5 == -2 -> {\n                val code =
codePointFrom2(bytes, byte, byteIndex, endIndex, throwOnMalformed)\n                if (code <= 0) {\n
stringBuilder.append(REPLACEMENT_CHAR)\n                    byteIndex += -code\n                } else {\n
stringBuilder.append(code.toChar())\n                    byteIndex += 1\n                }\n            }\n            byte shr 4 == -2 -
> {\n                val code = codePointFrom3(bytes, byte, byteIndex, endIndex, throwOnMalformed)\n                if
(code <= 0) {\n                    stringBuilder.append(REPLACEMENT_CHAR)\n                    byteIndex += -code\n
                } else {\n                    stringBuilder.append(code.toChar())\n                    byteIndex += 2\n
                }\n            }\n            byte shr 3 == -2 -> {\n                val code = codePointFrom4(bytes, byte, byteIndex, endIndex,
throwOnMalformed)\n                if (code <= 0) {\n                    stringBuilder.append(REPLACEMENT_CHAR)\n
                    byteIndex += -code\n                } else {\n                    val high = (code - 0x10000) shr 10 or 0xD800\n
                    val low = (code and 0x3FF) or 0xDC00\n                    stringBuilder.append(high.toChar())\n
stringBuilder.append(low.toChar())\n                    byteIndex += 3\n                }\n            }\n            else -> {\n
                malformed(0, byteIndex, throwOnMalformed)\n                stringBuilder.append(REPLACEMENT_CHAR)\n
            }\n        }\n    }\n    return stringBuilder.toString()\n}\n\n"/**\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin
Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.\n */\n\npackage kotlin\n\n/**\n * Returns the detailed description of this
throwable with its stack trace.\n */\n * The detailed description includes:\n * - the short description (see
[Throwable.toString]) of this throwable;\n * - the complete stack trace;\n * - detailed descriptions of the exceptions
that were [suppressed][suppressedExceptions] in order to deliver this exception;\n * - the detailed description of
each throwable in the [Throwable.cause] chain.\n */\n\n@SinceKotlin("1.4")\npublic actual fun
Throwable.stackTraceToString(): String = ExceptionTraceBuilder().buildFor(this)\n\n/**\n * Prints the [detailed
description][Throwable.stackTraceToString] of this throwable to console error output.\n
*/\n\n@SinceKotlin("1.4")\npublic actual fun Throwable.printStackTrace() {\n
    console.error(this.stackTraceToString())\n}\n\n/**\n * Adds the specified exception to the list of exceptions that
were\n * suppressed in order to deliver this exception.\n */\n\n@SinceKotlin("1.4")\npublic actual fun
Throwable.addSuppressed(exception: Throwable) {\n    if (this !== exception) {\n        val suppressed =
this.asDynamic()._suppressed.unsafeCast<MutableList<Throwable>?>()\n        if (suppressed == null) {\n
            this.asDynamic()._suppressed = mutableListOf(exception)\n        } else {\n            suppressed.add(exception)\n
        }\n    }\n}\n\n/**\n * Returns a list of all exceptions that were suppressed in order to deliver this exception.\n
*/\n\n@SinceKotlin("1.4")\npublic actual val Throwable.suppressedExceptions: List<Throwable>\n    get() {\n
    return this.asDynamic()._suppressed?.unsafeCast<List<Throwable>>() ?: emptyList()\n    }\n}\n\nprivate class
ExceptionTraceBuilder {\n    private val target = StringBuilder()\n    private val visited = arrayOf<Throwable>()\n
    private var topStack: String = ""\n    private var topStackStart: Int = 0\n    fun buildFor(exception: Throwable):
String {\n        exception.dumpFullTrace("\\", "\\")\n        return target.toString()\n    }\n    private fun

```

```

hasSeen(exception: Throwable): Boolean = visited.any { it === exception }\n\n private fun
Throwable.dumpFullTrace(indent: String, qualifier: String) {\n    this.dumpSelfTrace(indent, qualifier) ||
return\n\n    var cause = this.cause\n    while (cause != null) {\n        cause.dumpSelfTrace(indent, \"Caused
by: \") || return\n        cause = cause.cause\n    }\n}\n\n private fun Throwable.dumpSelfTrace(indent:
String, qualifier: String): Boolean {\n    target.append(indent).append(qualifier)\n    val shortInfo =
this.toString()\n    if (hasSeen(this)) {\n        target.append(\"[CIRCULAR REFERENCE, SEE ABOVE:
\"]) .append(shortInfo).append(\"\\n\\n\")\n        return false\n    }\n    visited.asDynamic().push(this)\n    var
stack = this.asDynamic().stack as String?\n    if (stack != null) {\n        val stackStart =
stack.indexOf(shortInfo).let { if (it < 0) 0 else it + shortInfo.length }\n        if (stackStart == 0)
target.append(shortInfo).append(\"\\n\\n\")\n        if (topStack.isEmpty()) {\n            topStack = stack\n
topStackStart = stackStart\n        } else {\n            stack = dropCommonFrames(stack, stackStart)\n        }\n
        if (indent.isNotEmpty()) {\n            // indent stack, but avoid indenting exception message lines\n            val
messageLines = if (stackStart == 0) 0 else 1 + shortInfo.count { c -> c == '\\n' }\n
stack.lineSequence().forEachIndexed { index: Int, line: String -> }\n            if (index >= messageLines)
target.append(indent)\n            target.append(line).append(\"\\n\\n\")\n        } else {\n            target.append(stack).append(\"\\n\\n\")\n        }\n    } else {\n        target.append(shortInfo).append(\"\\n\\n\")\n    }\n\n    val suppressed = suppressedExceptions\n    if (suppressed.isNotEmpty()) {\n        val
suppressedIndent = indent + \"  \"\n        for (s in suppressed) {\n            s.dumpFullTrace(suppressedIndent,
\"Suppressed: \")\n        }\n    }\n    return true\n}\n\n private fun dropCommonFrames(stack: String,
stackStart: Int): String {\n    var commonFrames: Int = 0\n    var lastBreak: Int = 0\n    var preLastBreak: Int
= 0\n    for (pos in 0 until minOf(topStack.length - topStackStart, stack.length - stackStart)) {\n        val c =
stack[stack.lastIndex - pos]\n        if (c != topStack[topStack.lastIndex - pos]) break\n        if (c == '\\n') {\n
            commonFrames += 1\n            preLastBreak = lastBreak\n            lastBreak = pos\n        }\n    }\n
    if (commonFrames <= 1) return stack\n    while (preLastBreak > 0 && stack[stack.lastIndex - (preLastBreak - 1)]
== ' ')\n        preLastBreak -= 1\n    // leave 1 common frame to ease matching with the top exception stack\n
    return stack.dropLast(preLastBreak) + \"... and ${commonFrames - 1} more common stack frames skipped\"\n}\n\n
/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this
source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*\n\npackage kotlin.time\nimport kotlin.js.json\nimport kotlin.math.*\n\ninternal actual inline val
durationAssertionsEnabled: Boolean get() = true\n\ninternal actual fun formatToExactDecimals(value: Double,
decimals: Int): String {\n    val rounded = if (decimals == 0) {\n        value\n    } else {\n        val pow =
10.0.pow(decimals)\n        JsMath.round(abs(value) * pow) / pow * sign(value)\n    }\n    return if (abs(rounded) <
1e21) {\n        // toFixed switches to scientific format after 1e21\n
rounded.asDynamic().toFixed(decimals).unsafeCast<String>()\n    } else {\n        // toPrecision outputs the specified
number of digits, but only for positive numbers\n        val positive = abs(rounded)\n        val positiveString =
positive.asDynamic().toPrecision(ceil(log10(positive)) + decimals).unsafeCast<String>()\n        if (rounded < 0) \"-
$positiveString\" else positiveString\n    }\n}\n\ninternal actual fun formatUpToDecimals(value: Double, decimals:
Int): String {\n    return value.asDynamic().toLocaleString(\"en-us\", json(\"maximumFractionDigits\" to
decimals)).unsafeCast<String>()\n}\n\n/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\npackage
kotlin.time\n@SinceKotlin(\"1.6\")\n@WasExperimental(ExperimentalTime::class)\npublic actual enum class
DurationUnit(internal val scale: Double) {\n    /**\n     * Time unit representing one nanosecond, which is 1/1000
of a microsecond.\n     *\n     * NANOSECONDS(1e0),\n     /**\n     * Time unit representing one microsecond, which is
1/1000 of a millisecond.\n     *\n     * MICROSECONDS(1e3),\n     /**\n     * Time unit representing one millisecond,
which is 1/1000 of a second.\n     *\n     * MILLISECONDS(1e6),\n     /**\n     * Time unit representing one second.\n
     *\n     * SECONDS(1e9),\n     /**\n     * Time unit representing one minute.\n     *\n     * MINUTES(60e9),\n     /**\n
     * Time unit representing one hour.\n     *\n     * HOURS(3600e9),\n     /**\n     * Time unit representing one day,

```

```

which is always equal to 24 hours.\n    *^\n    DAYS(86400e9);\n}\n\n@SinceKotlin("1.3")\ninternal actual fun
convertDurationUnit(value: Double, sourceUnit: DurationUnit, targetUnit: DurationUnit): Double {\n    val
sourceCompareTarget = sourceUnit.scale.compareTo(targetUnit.scale)\n    return when {\n
sourceCompareTarget > 0 -> value * (sourceUnit.scale / targetUnit.scale)\n    sourceCompareTarget < 0 -> value /
(targetUnit.scale / sourceUnit.scale)\n    else -> value\n    }\n}\n\n@SinceKotlin("1.5")\ninternal actual fun
convertDurationUnitOverflow(value: Long, sourceUnit: DurationUnit, targetUnit: DurationUnit): Long {\n    val
sourceCompareTarget = sourceUnit.scale.compareTo(targetUnit.scale)\n    return when {\n
sourceCompareTarget > 0 -> value * (sourceUnit.scale / targetUnit.scale).toLong()\n    sourceCompareTarget < 0
-> value / (targetUnit.scale / sourceUnit.scale).toLong()\n    else -> value\n
}\n}\n\n@SinceKotlin("1.5")\ninternal actual fun convertDurationUnit(value: Long, sourceUnit: DurationUnit,
targetUnit: DurationUnit): Long {\n    val sourceCompareTarget = sourceUnit.scale.compareTo(targetUnit.scale)\n
return when {\n    sourceCompareTarget > 0 -> {\n        val scale = (sourceUnit.scale /
targetUnit.scale).toLong()\n        val result = value * scale\n        when {\n            result / scale == value ->
result\n            value > 0 -> Long.MAX_VALUE\n            else -> Long.MIN_VALUE\n        }\n    }\n
sourceCompareTarget < 0 -> value / (targetUnit.scale / sourceUnit.scale).toLong()\n    else -> value\n
}\n}\n\n\n", /*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of
this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*\n *\npackage kotlin.time\nimport org.w3c.performance.GlobalPerformance\nimport
org.w3c.performance.Performance\n\n@SinceKotlin("1.3")\n@ExperimentalTime\ninternal actual object
MonotonicTimeSource : TimeSource {\n    private val actualSource: TimeSource = run {\n        val isNode:
Boolean = js("typeof process !== 'undefined' && process.versions && !process.versions.node")\n        if
(isNode)\n            HrTimeSource(js("process").unsafeCast<Process>())\n        else\n
js("self").unsafeCast<GlobalPerformance?>()?.performance?.let(::PerformanceTimeSource)\n        ?:
DateNowTimeSource\n    }\n    override fun markNow(): TimeMark = actualSource.markNow()\n}\n\ninternal
external interface Process {\n    fun hrtime(time: Array<Double> = definedExternally):
Array<Double>\n}\n\n@SinceKotlin("1.3")\n@ExperimentalTime\ninternal class HrTimeSource(val process:
Process) : TimeSource {\n    override fun markNow(): TimeMark = object : TimeMark() {\n        val startedAt =
process.hrtime()\n        override fun elapsedNow(): Duration =\n            process.hrtime(startedAt).let { (seconds,
nanos) -> seconds.toDuration(DurationUnit.SECONDS) + nanos.toDuration(DurationUnit.NANOSECONDS) }\n
}\n    override fun toString(): String =
\n"TimeSource(process.hrtime())"\n}\n\n@SinceKotlin("1.3")\n@ExperimentalTime\ninternal class
PerformanceTimeSource(val performance: Performance) : AbstractDoubleTimeSource(unit =
DurationUnit.MILLISECONDS) {\n    override fun read(): Double = performance.now()\n    override fun toString():
String = \n"TimeSource(self.performance.now())"\n}\n\n@SinceKotlin("1.3")\n@ExperimentalTime\ninternal
object DateNowTimeSource : AbstractDoubleTimeSource(unit = DurationUnit.MILLISECONDS) {\n    override
fun read(): Double = kotlin.js.Date.now()\n    override fun toString(): String = \n"TimeSource(Date.now())"\n}, /*\n
*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code
is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n *\n *\npackage
kotlinx.dom\nimport org.w3c.dom.*\nimport kotlin.contracts.*\n\n * Creates a new element with the
specified [name].\n * The element is initialized with the specified [init] function.\n
*\n\n@SinceKotlin("1.4")\npublic fun Document.createElement(name: String, init: Element.() -> Unit): Element {\n
    contract { callsInPlace(init, InvocationKind.EXACTLY_ONCE) }\n    return
createElement(name).apply(init)\n}\n\n * Appends a newly created element with the specified [name] to this
element.\n * The element is initialized with the specified [init] function.\n *\n\n@SinceKotlin("1.4")\npublic fun
Element.appendChild(name: String, init: Element.() -> Unit): Element {\n    contract { callsInPlace(init,
InvocationKind.EXACTLY_ONCE) }\n    return ownerDocument!!.createElement(name, init).also {
appendChild(it) }\n}\n\n", /*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the

```

license/LICENSE.txt file.\n \*/\n\npackage kotlin.dom\n\nimport org.w3c.dom.\*\n\n/\*\* Returns true if the element has the given CSS class style in its 'class' attribute \*/\n@SinceKotlin("1.4")\nfun Element.hasClass(cssClass: String): Boolean = className.matches("\\\\\\"(\\^.\*\\|s+)\$cssClass(\\|s+.\*)"\\|").toRegex())\n\n/\*\*\n \* Adds CSS class to element. Has no effect if all specified classes are already in class attribute of the element\n \*/\n \* @return true if at least one class has been added\n \*/\n@SinceKotlin("1.4")\nfun Element.addClass(vararg cssClasses: String): Boolean {\n val missingClasses = cssClasses.filterNot { hasClass(it) }\n if (missingClasses.isNotEmpty()) {\n val presentClasses = className.trim()\n className = buildString {\n append(presentClasses)\n if (!presentClasses.isEmpty()) {\n append(" ")\n }\n missingClasses.joinTo(this, " ")\n }\n return true\n }\n return false\n}\n\n/\*\*\n \* Removes all [cssClasses] from element. Has no effect if all specified classes are missing in class attribute of the element\n \*/\n \* @return true if at least one class has been removed\n \*/\n@SinceKotlin("1.4")\nfun Element.removeClass(vararg cssClasses: String): Boolean {\n if (cssClasses.any { hasClass(it) }) {\n val toBeRemoved = cssClasses.toSet()\n className = className.trim().split("\\\\\\"s+"\\|").toRegex().filter { it !in toBeRemoved }.joinToString(" ")\n return true\n }\n return false\n}\n\n", /\*\n \* Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n \* Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n

```

*/\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("StringsKt")\n\npackage kotlin.text\n\n/**\n * Converts the string into a regular expression [Regex] with the default options.\n */\n@kotlin.internal.InlineOnly\npublic inline fun String.toRegex(): Regex = Regex(this)\n\n/**\n * Converts the string into a regular expression [Regex] with the specified single [option].\n */\n@kotlin.internal.InlineOnly\npublic inline fun String.toRegex(option: RegexOption): Regex = Regex(this, option)\n\n/**\n * Converts the string into a regular expression [Regex] with the specified set of [options].\n */\n@kotlin.internal.InlineOnly\npublic inline fun String.toRegex(options: Set<RegexOption>): Regex = Regex(this, options)\n\n", /*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.dom\n\nimport org.w3c.dom.*\n\n/**\n * Gets a value indicating whether this node is a TEXT_NODE or a CDATA_SECTION_NODE.\n */\n@SinceKotlin("1.4")\npublic val Node.isText: Boolean\n    get() = nodeType == Node.TEXT_NODE || nodeType == Node.CDATA_SECTION_NODE\n\n/**\n * Gets a value indicating whether this node is an [Element].\n */\n@SinceKotlin("1.4")\npublic val Node.isElement: Boolean\n    get() = nodeType == Node.ELEMENT_NODE\n\n", /*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.dom\n\nimport org.w3c.dom.*\n\n/**\n * Removes all the children from this node.\n */\n@SinceKotlin("1.4")\npublic fun Node.clear() {\n    while (hasChildNodes()) {\n        removeChild(firstChild!)\n    }\n}\n\n/**\n * Creates text node and append it to the element.\n */\n * @return this element\n */\n@SinceKotlin("1.4")\nfun Element.appendText(text: String): Element {\n    appendChild(ownerDocument!!.createTextNode(text))\n    return this\n}\n\n", /*\n * Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage org.w3c.dom\n\n@Deprecated("Use UnionMessagePortOrWindowProxy instead.")\nReplaceWith("UnionMessagePortOrWindowProxy")\n\ninterface UnionMessagePortOrWindow {\n    @Deprecated("Use `as` instead.")\n    ReplaceWith("`as`")\n    var HTMLLinkElement.as_: Boolean\n    get() = `as`\n    set(value) {\n        `as` = value\n    }\n}\n\n@Deprecated("Use `is` instead.")\nReplaceWith("`is`")\n\ninterface ElementCreationOptions {\n    is_: Boolean\n    get() = `is`\n    set(value) {\n        `is` = value\n    }\n}\n\n", /*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n// NOTE: THIS FILE IS AUTO-GENERATED, DO NOT EDIT!\n// See github.com/kotlin/dukat for details\n\npackage org.khronos.webgl\n\nimport kotlin.js.*\nimport org.w3c.dom.*\nimport org.w3c.dom.events.*\n\npublic external interface WebGLContextAttributes {\n    var alpha: Boolean? /* = true */\n    get() = definedExternally\n    set(value) = definedExternally\n    var depth:

```

```

Boolean? /* = true */\n    get() = definedExternally\n    set(value) = definedExternally\n    var stencil: Boolean?
/* = false */\n    get() = definedExternally\n    set(value) = definedExternally\n    var antialias: Boolean? /* =
true */\n    get() = definedExternally\n    set(value) = definedExternally\n    var premultipliedAlpha: Boolean?
/* = true */\n    get() = definedExternally\n    set(value) = definedExternally\n    var preserveDrawingBuffer:
Boolean? /* = false */\n    get() = definedExternally\n    set(value) = definedExternally\n    var
preferLowPowerToHighPerformance: Boolean? /* = false */\n    get() = definedExternally\n    set(value) =
definedExternally\n    var failIfMajorPerformanceCaveat: Boolean? /* = false */\n    get() = definedExternally\n
    set(value) = definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun WebGLContextAttributes(alpha:
Boolean? = true, depth: Boolean? = true, stencil: Boolean? = false, antialias: Boolean? = true, premultipliedAlpha:
Boolean? = true, preserveDrawingBuffer: Boolean? = false, preferLowPowerToHighPerformance: Boolean? = false,
failIfMajorPerformanceCaveat: Boolean? = false): WebGLContextAttributes {\n    val o = js("{}")\n    o["alpha"] = alpha\n    o["depth"] = depth\n    o["stencil"] = stencil\n    o["antialias"] = antialias\n
    o["premultipliedAlpha"] = premultipliedAlpha\n    o["preserveDrawingBuffer"] = preserveDrawingBuffer\n
    o["preferLowPowerToHighPerformance"] = preferLowPowerToHighPerformance\n
    o["failIfMajorPerformanceCaveat"] = failIfMajorPerformanceCaveat\n    return o\n}\n\npublic external abstract
class WebGLObject\n\n/**\n * Exposes the JavaScript
[WebGLBuffer](https://developer.mozilla.org/en/docs/Web/API/WebGLBuffer) to Kotlin\n */\n\npublic external
abstract class WebGLBuffer : WebGLObject\n\n/**\n * Exposes the JavaScript
[WebGLFramebuffer](https://developer.mozilla.org/en/docs/Web/API/WebGLFramebuffer) to Kotlin\n */\n\npublic
external abstract class WebGLFramebuffer : WebGLObject\n\n/**\n * Exposes the JavaScript
[WebGLProgram](https://developer.mozilla.org/en/docs/Web/API/WebGLProgram) to Kotlin\n */\n\npublic external
abstract class WebGLProgram : WebGLObject\n\n/**\n * Exposes the JavaScript
[WebGLRenderbuffer](https://developer.mozilla.org/en/docs/Web/API/WebGLRenderbuffer) to Kotlin\n */\n\npublic
external abstract class WebGLRenderbuffer : WebGLObject\n\n/**\n * Exposes the JavaScript
[WebGLShader](https://developer.mozilla.org/en/docs/Web/API/WebGLShader) to Kotlin\n */\n\npublic external
abstract class WebGLShader : WebGLObject\n\n/**\n * Exposes the JavaScript
[WebGLTexture](https://developer.mozilla.org/en/docs/Web/API/WebGLTexture) to Kotlin\n */\n\npublic external
abstract class WebGLTexture : WebGLObject\n\n/**\n * Exposes the JavaScript
[WebGLUniformLocation](https://developer.mozilla.org/en/docs/Web/API/WebGLUniformLocation) to Kotlin\n
*/\n\npublic external abstract class WebGLUniformLocation\n\n/**\n * Exposes the JavaScript
[WebGLActiveInfo](https://developer.mozilla.org/en/docs/Web/API/WebGLActiveInfo) to Kotlin\n */\n\npublic
external abstract class WebGLActiveInfo {\n    open val size: Int\n    open val type: Int\n    open val name:
String\n}\n\n/**\n * Exposes the JavaScript
[WebGLShaderPrecisionFormat](https://developer.mozilla.org/en/docs/Web/API/WebGLShaderPrecisionFormat) to
Kotlin\n */\n\npublic external abstract class WebGLShaderPrecisionFormat {\n    open val rangeMin: Int\n    open val
rangeMax: Int\n    open val precision:
Int\n}\n\n\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\n\npublic external interface
WebGLRenderingContextBase {\n    val canvas: HTMLCanvasElement\n    val drawingBufferWidth: Int\n    val
drawingBufferHeight: Int\n    fun getContextAttributes(): WebGLContextAttributes?\n    fun isContextLost():
Boolean\n    fun getSupportedExtensions(): Array<String>?\n    fun getExtension(name: String): dynamic\n    fun
activeTexture(texture: Int)\n    fun attachShader(program: WebGLProgram?, shader: WebGLShader?)\n    fun
bindAttribLocation(program: WebGLProgram?, index: Int, name: String)\n    fun bindBuffer(target: Int, buffer:
WebGLBuffer?)\n    fun bindFramebuffer(target: Int, framebuffer: WebGLFramebuffer?)\n    fun
bindRenderbuffer(target: Int, renderbuffer: WebGLRenderbuffer?)\n    fun bindTexture(target: Int, texture:
WebGLTexture?)\n    fun blendColor(red: Float, green: Float, blue: Float, alpha: Float)\n    fun
blendEquation(mode: Int)\n    fun blendEquationSeparate(modeRGB: Int, modeAlpha: Int)\n    fun
blendFunc(sfactor: Int, dfactor: Int)\n    fun blendFuncSeparate(srcRGB: Int, dstRGB: Int, srcAlpha: Int, dstAlpha:
Int)

```

Int)\n fun bufferData(target: Int, size: Int, usage: Int)\n fun bufferData(target: Int, data: BufferDataSource?, usage: Int)\n fun bufferSubData(target: Int, offset: Int, data: BufferDataSource?)\n fun checkFramebufferStatus(target: Int): Int\n fun clear(mask: Int)\n fun clearColor(red: Float, green: Float, blue: Float, alpha: Float)\n fun clearDepth(depth: Float)\n fun clearStencil(s: Int)\n fun colorMask(red: Boolean, green: Boolean, blue: Boolean, alpha: Boolean)\n fun compileShader(shader: WebGLShader?)\n fun compressedTexImage2D(target: Int, level: Int, internalformat: Int, width: Int, height: Int, border: Int, data: ArrayBufferView)\n fun compressedTexSubImage2D(target: Int, level: Int, xoffset: Int, yoffset: Int, width: Int, height: Int, format: Int, data: ArrayBufferView)\n fun copyTexImage2D(target: Int, level: Int, internalformat: Int, x: Int, y: Int, width: Int, height: Int, border: Int)\n fun copyTexSubImage2D(target: Int, level: Int, xoffset: Int, yoffset: Int, x: Int, y: Int, width: Int, height: Int)\n fun createBuffer(): WebGLBuffer?\n fun createFramebuffer(): WebGLFramebuffer?\n fun createProgram(): WebGLProgram?\n fun createRenderbuffer(): WebGLRenderbuffer?\n fun createShader(type: Int): WebGLShader?\n fun createTexture(): WebGLTexture?\n fun cullFace(mode: Int)\n fun deleteBuffer(buffer: WebGLBuffer?)\n fun deleteFramebuffer(framebuffer: WebGLFramebuffer?)\n fun deleteProgram(program: WebGLProgram?)\n fun deleteRenderbuffer(renderbuffer: WebGLRenderbuffer?)\n fun deleteShader(shader: WebGLShader?)\n fun deleteTexture(texture: WebGLTexture?)\n fun depthFunc(func: Int)\n fun depthMask(flag: Boolean)\n fun depthRange(zNear: Float, zFar: Float)\n fun detachShader(program: WebGLProgram?, shader: WebGLShader?)\n fun disable(cap: Int)\n fun disableVertexArray(index: Int)\n fun drawArrays(mode: Int, first: Int, count: Int)\n fun drawElements(mode: Int, count: Int, type: Int, offset: Int)\n fun enable(cap: Int)\n fun enableVertexArray(index: Int)\n fun finish()\n fun flush()\n fun framebufferRenderbuffer(target: Int, attachment: Int, renderbuffertarget: Int, renderbuffer: WebGLRenderbuffer?)\n fun framebufferTexture2D(target: Int, attachment: Int, textarget: Int, texture: WebGLTexture?, level: Int)\n fun frontFace(mode: Int)\n fun generateMipmap(target: Int)\n fun getActiveAttrib(program: WebGLProgram?, index: Int): WebGLActiveInfo?\n fun getActiveUniform(program: WebGLProgram?, index: Int): WebGLActiveInfo?\n fun getAttachedShaders(program: WebGLProgram?): Array<WebGLShader>?\n fun getAttribLocation(program: WebGLProgram?, name: String): Int\n fun getBufferParameter(target: Int, pname: Int): Any?\n fun getParameter(pname: Int): Any?\n fun getError(): Int\n fun getFramebufferAttachmentParameter(target: Int, attachment: Int, pname: Int): Any?\n fun getProgramParameter(program: WebGLProgram?, pname: Int): Any?\n fun getProgramInfoLog(program: WebGLProgram?): String?\n fun getRenderbufferParameter(target: Int, pname: Int): Any?\n fun getShaderParameter(shader: WebGLShader?, pname: Int): Any?\n fun getShaderPrecisionFormat(shadertype: Int, precisiontype: Int): WebGLShaderPrecisionFormat?\n fun getShaderInfoLog(shader: WebGLShader?): String?\n fun getShaderSource(shader: WebGLShader?): String?\n fun getTexParameter(target: Int, pname: Int): Any?\n fun getUniform(program: WebGLProgram?, location: WebGLUniformLocation?): Any?\n fun getUniformLocation(program: WebGLProgram?, name: String): WebGLUniformLocation?\n fun getVertexAttrib(index: Int, pname: Int): Any?\n fun getVertexAttribOffset(index: Int, pname: Int): Int\n fun hint(target: Int, mode: Int)\n fun isBuffer(buffer: WebGLBuffer?): Boolean\n fun isEnabled(cap: Int): Boolean\n fun isFramebuffer(framebuffer: WebGLFramebuffer?): Boolean\n fun isProgram(program: WebGLProgram?): Boolean\n fun isRenderbuffer(renderbuffer: WebGLRenderbuffer?): Boolean\n fun isShader(shader: WebGLShader?): Boolean\n fun isTexture(texture: WebGLTexture?): Boolean\n fun lineWidth(width: Float)\n fun linkProgram(program: WebGLProgram?)\n fun pixelStorei(pname: Int, param: Int)\n fun polygonOffset(factor: Float, units: Float)\n fun readPixels(x: Int, y: Int, width: Int, height: Int, format: Int, type: Int, pixels: ArrayBufferView?)\n fun renderbufferStorage(target: Int, internalformat: Int, width: Int, height: Int)\n fun sampleCoverage(value: Float, invert: Boolean)\n fun scissor(x: Int, y: Int, width: Int, height: Int)\n fun shaderSource(shader: WebGLShader?, source: String)\n fun stencilFunc(func: Int, ref: Int, mask: Int)\n fun stencilFuncSeparate(face: Int, func: Int, ref: Int, mask: Int)\n fun stencilMask(mask: Int)\n fun stencilMaskSeparate(face: Int, mask: Int)\n fun stencilOp(fail: Int, zfail: Int, zpass: Int)\n fun stencilOpSeparate(face: Int, fail: Int, zfail: Int, zpass: Int)\n fun texImage2D(target: Int, level: Int, internalformat: Int, width: Int, height: Int, border: Int, format: Int, type: Int, pixels:

```

ArrayBufferView?)\n fun texImage2D(target: Int, level: Int, internalformat: Int, format: Int, type: Int, source:
TexImageSource?)\n fun texParameterf(target: Int, pname: Int, param: Float)\n fun texParameteri(target: Int,
pname: Int, param: Int)\n fun texSubImage2D(target: Int, level: Int, xoffset: Int, yoffset: Int, width: Int, height: Int,
format: Int, type: Int, pixels: ArrayBufferView?)\n fun texSubImage2D(target: Int, level: Int, xoffset: Int, yoffset:
Int, format: Int, type: Int, source: TexImageSource?)\n fun uniform1f(location: WebGLUniformLocation?, x:
Float)\n fun uniform1fv(location: WebGLUniformLocation?, v: Float32Array)\n fun uniform1fv(location:
WebGLUniformLocation?, v: Array<Float>)\n fun uniform1i(location: WebGLUniformLocation?, x: Int)\n fun
uniform1iv(location: WebGLUniformLocation?, v: Int32Array)\n fun uniform1iv(location:
WebGLUniformLocation?, v: Array<Int>)\n fun uniform2f(location: WebGLUniformLocation?, x: Float, y:
Float)\n fun uniform2fv(location: WebGLUniformLocation?, v: Float32Array)\n fun uniform2fv(location:
WebGLUniformLocation?, v: Array<Float>)\n fun uniform2i(location: WebGLUniformLocation?, x: Int, y: Int)\n
fun uniform2iv(location: WebGLUniformLocation?, v: Int32Array)\n fun uniform2iv(location:
WebGLUniformLocation?, v: Array<Int>)\n fun uniform3f(location: WebGLUniformLocation?, x: Float, y: Float,
z: Float)\n fun uniform3fv(location: WebGLUniformLocation?, v: Float32Array)\n fun uniform3fv(location:
WebGLUniformLocation?, v: Array<Float>)\n fun uniform3i(location: WebGLUniformLocation?, x: Int, y: Int, z:
Int)\n fun uniform3iv(location: WebGLUniformLocation?, v: Int32Array)\n fun uniform3iv(location:
WebGLUniformLocation?, v: Array<Int>)\n fun uniform4f(location: WebGLUniformLocation?, x: Float, y: Float,
z: Float, w: Float)\n fun uniform4fv(location: WebGLUniformLocation?, v: Float32Array)\n fun
uniform4fv(location: WebGLUniformLocation?, v: Array<Float>)\n fun uniform4i(location:
WebGLUniformLocation?, x: Int, y: Int, z: Int, w: Int)\n fun uniform4iv(location: WebGLUniformLocation?, v:
Int32Array)\n fun uniform4iv(location: WebGLUniformLocation?, v: Array<Int>)\n fun
uniformMatrix2fv(location: WebGLUniformLocation?, transpose: Boolean, value: Float32Array)\n fun
uniformMatrix2fv(location: WebGLUniformLocation?, transpose: Boolean, value: Array<Float>)\n fun
uniformMatrix3fv(location: WebGLUniformLocation?, transpose: Boolean, value: Float32Array)\n fun
uniformMatrix3fv(location: WebGLUniformLocation?, transpose: Boolean, value: Array<Float>)\n fun
uniformMatrix4fv(location: WebGLUniformLocation?, transpose: Boolean, value: Float32Array)\n fun
uniformMatrix4fv(location: WebGLUniformLocation?, transpose: Boolean, value: Array<Float>)\n fun
useProgram(program: WebGLProgram?)\n fun validateProgram(program: WebGLProgram?)\n fun
vertexAttrib1f(index: Int, x: Float)\n fun vertexAttrib1fv(index: Int, values: dynamic)\n fun
vertexAttrib2f(index: Int, x: Float, y: Float)\n fun vertexAttrib2fv(index: Int, values: dynamic)\n fun
vertexAttrib3f(index: Int, x: Float, y: Float, z: Float)\n fun vertexAttrib3fv(index: Int, values: dynamic)\n fun
vertexAttrib4f(index: Int, x: Float, y: Float, z: Float, w: Float)\n fun vertexAttrib4fv(index: Int, values: dynamic)\n
fun vertexAttribPointer(index: Int, size: Int, type: Int, normalized: Boolean, stride: Int, offset: Int)\n fun
viewport(x: Int, y: Int, width: Int, height: Int)\n\n companion object {\n val DEPTH_BUFFER_BIT: Int\n
val STENCIL_BUFFER_BIT: Int\n val COLOR_BUFFER_BIT: Int\n val POINTS: Int\n val LINES:
Int\n val LINE_LOOP: Int\n val LINE_STRIP: Int\n val TRIANGLES: Int\n val
TRIANGLE_STRIP: Int\n val TRIANGLE_FAN: Int\n val ZERO: Int\n val ONE: Int\n val
SRC_COLOR: Int\n val ONE_MINUS_SRC_COLOR: Int\n val SRC_ALPHA: Int\n val
ONE_MINUS_SRC_ALPHA: Int\n val DST_ALPHA: Int\n val ONE_MINUS_DST_ALPHA: Int\n
val DST_COLOR: Int\n val ONE_MINUS_DST_COLOR: Int\n val SRC_ALPHA_SATURATE: Int\n
val FUNC_ADD: Int\n val BLEND_EQUATION: Int\n val BLEND_EQUATION_RGB: Int\n val
BLEND_EQUATION_ALPHA: Int\n val FUNC_SUBTRACT: Int\n val FUNC_REVERSE_SUBTRACT:
Int\n val BLEND_DST_RGB: Int\n val BLEND_SRC_RGB: Int\n val BLEND_DST_ALPHA: Int\n
val BLEND_SRC_ALPHA: Int\n val CONSTANT_COLOR: Int\n val
ONE_MINUS_CONSTANT_COLOR: Int\n val CONSTANT_ALPHA: Int\n val
ONE_MINUS_CONSTANT_ALPHA: Int\n val BLEND_COLOR: Int\n val ARRAY_BUFFER: Int\n
val ELEMENT_ARRAY_BUFFER: Int\n val ARRAY_BUFFER_BINDING: Int\n val
ELEMENT_ARRAY_BUFFER_BINDING: Int\n val STREAM_DRAW: Int\n val STATIC_DRAW: Int\n

```

val DYNAMIC\_DRAW: Int\n    val BUFFER\_SIZE: Int\n    val BUFFER\_USAGE: Int\n    val  
 CURRENT\_VERTEX\_ATTRIB: Int\n    val FRONT: Int\n    val BACK: Int\n    val FRONT\_AND\_BACK:  
 Int\n    val CULL\_FACE: Int\n    val BLEND: Int\n    val DITHER: Int\n    val STENCIL\_TEST: Int\n  
 val DEPTH\_TEST: Int\n    val SCISSOR\_TEST: Int\n    val POLYGON\_OFFSET\_FILL: Int\n    val  
 SAMPLE\_ALPHA\_TO\_COVERAGE: Int\n    val SAMPLE\_COVERAGE: Int\n    val NO\_ERROR: Int\n  
 val INVALID\_ENUM: Int\n    val INVALID\_VALUE: Int\n    val INVALID\_OPERATION: Int\n    val  
 OUT\_OF\_MEMORY: Int\n    val CW: Int\n    val CCW: Int\n    val LINE\_WIDTH: Int\n    val  
 ALIASED\_POINT\_SIZE\_RANGE: Int\n    val ALIASED\_LINE\_WIDTH\_RANGE: Int\n    val  
 CULL\_FACE\_MODE: Int\n    val FRONT\_FACE: Int\n    val DEPTH\_RANGE: Int\n    val  
 DEPTH\_WRITEMASK: Int\n    val DEPTH\_CLEAR\_VALUE: Int\n    val DEPTH\_FUNC: Int\n    val  
 STENCIL\_CLEAR\_VALUE: Int\n    val STENCIL\_FUNC: Int\n    val STENCIL\_FAIL: Int\n    val  
 STENCIL\_PASS\_DEPTH\_FAIL: Int\n    val STENCIL\_PASS\_DEPTH\_PASS: Int\n    val STENCIL\_REF:  
 Int\n    val STENCIL\_VALUE\_MASK: Int\n    val STENCIL\_WRITEMASK: Int\n    val  
 STENCIL\_BACK\_FUNC: Int\n    val STENCIL\_BACK\_FAIL: Int\n    val  
 STENCIL\_BACK\_PASS\_DEPTH\_FAIL: Int\n    val STENCIL\_BACK\_PASS\_DEPTH\_PASS: Int\n    val  
 STENCIL\_BACK\_REF: Int\n    val STENCIL\_BACK\_VALUE\_MASK: Int\n    val  
 STENCIL\_BACK\_WRITEMASK: Int\n    val VIEWPORT: Int\n    val SCISSOR\_BOX: Int\n    val  
 COLOR\_CLEAR\_VALUE: Int\n    val COLOR\_WRITEMASK: Int\n    val UNPACK\_ALIGNMENT: Int\n  
 val PACK\_ALIGNMENT: Int\n    val MAX\_TEXTURE\_SIZE: Int\n    val MAX\_VIEWPORT\_DIMS: Int\n  
 val SUBPIXEL\_BITS: Int\n    val RED\_BITS: Int\n    val GREEN\_BITS: Int\n    val BLUE\_BITS: Int\n  
 val ALPHA\_BITS: Int\n    val DEPTH\_BITS: Int\n    val STENCIL\_BITS: Int\n    val  
 POLYGON\_OFFSET\_UNITS: Int\n    val POLYGON\_OFFSET\_FACTOR: Int\n    val  
 TEXTURE\_BINDING\_2D: Int\n    val SAMPLE\_BUFFERS: Int\n    val SAMPLES: Int\n    val  
 SAMPLE\_COVERAGE\_VALUE: Int\n    val SAMPLE\_COVERAGE\_INVERT: Int\n    val  
 COMPRESSED\_TEXTURE\_FORMATS: Int\n    val DONT\_CARE: Int\n    val FASTEST: Int\n    val  
 NICEST: Int\n    val GENERATE\_MIPMAP\_HINT: Int\n    val BYTE: Int\n    val UNSIGNED\_BYTE:  
 Int\n    val SHORT: Int\n    val UNSIGNED\_SHORT: Int\n    val INT: Int\n    val UNSIGNED\_INT: Int\n  
 val FLOAT: Int\n    val DEPTH\_COMPONENT: Int\n    val ALPHA: Int\n    val RGB: Int\n    val  
 RGBA: Int\n    val LUMINANCE: Int\n    val LUMINANCE\_ALPHA: Int\n    val  
 UNSIGNED\_SHORT\_4\_4\_4\_4: Int\n    val UNSIGNED\_SHORT\_5\_5\_5\_1: Int\n    val  
 UNSIGNED\_SHORT\_5\_6\_5: Int\n    val FRAGMENT\_SHADER: Int\n    val VERTEX\_SHADER: Int\n  
 val MAX\_VERTEX\_ATTRIBS: Int\n    val MAX\_VERTEX\_UNIFORM\_VECTORS: Int\n    val  
 MAX\_VARYING\_VECTORS: Int\n    val MAX\_COMBINED\_TEXTURE\_IMAGE\_UNITS: Int\n    val  
 MAX\_VERTEX\_TEXTURE\_IMAGE\_UNITS: Int\n    val MAX\_TEXTURE\_IMAGE\_UNITS: Int\n    val  
 MAX\_FRAGMENT\_UNIFORM\_VECTORS: Int\n    val SHADER\_TYPE: Int\n    val DELETE\_STATUS:  
 Int\n    val LINK\_STATUS: Int\n    val VALIDATE\_STATUS: Int\n    val ATTACHED\_SHADERS: Int\n  
 val ACTIVE\_UNIFORMS: Int\n    val ACTIVE\_ATTRIBUTES: Int\n    val  
 SHADING\_LANGUAGE\_VERSION: Int\n    val CURRENT\_PROGRAM: Int\n    val NEVER: Int\n    val  
 LESS: Int\n    val EQUAL: Int\n    val LEQUAL: Int\n    val GREATER: Int\n    val NOTEQUAL: Int\n  
 val GEQUAL: Int\n    val ALWAYS: Int\n    val KEEP: Int\n    val REPLACE: Int\n    val INCR: Int\n  
 val DECR: Int\n    val INVERT: Int\n    val INCR\_WRAP: Int\n    val DECR\_WRAP: Int\n    val  
 VENDOR: Int\n    val RENDERER: Int\n    val VERSION: Int\n    val NEAREST: Int\n    val LINEAR:  
 Int\n    val NEAREST\_MIPMAP\_NEAREST: Int\n    val LINEAR\_MIPMAP\_NEAREST: Int\n    val  
 NEAREST\_MIPMAP\_LINEAR: Int\n    val LINEAR\_MIPMAP\_LINEAR: Int\n    val  
 TEXTURE\_MAG\_FILTER: Int\n    val TEXTURE\_MIN\_FILTER: Int\n    val TEXTURE\_WRAP\_S: Int\n  
 val TEXTURE\_WRAP\_T: Int\n    val TEXTURE\_2D: Int\n    val TEXTURE: Int\n    val  
 TEXTURE\_CUBE\_MAP: Int\n    val TEXTURE\_BINDING\_CUBE\_MAP: Int\n    val  
 TEXTURE\_CUBE\_MAP\_POSITIVE\_X: Int\n    val TEXTURE\_CUBE\_MAP\_NEGATIVE\_X: Int\n    val

```

TEXTURE_CUBE_MAP_POSITIVE_Y: Int\n    val TEXTURE_CUBE_MAP_NEGATIVE_Y: Int\n    val
TEXTURE_CUBE_MAP_POSITIVE_Z: Int\n    val TEXTURE_CUBE_MAP_NEGATIVE_Z: Int\n    val
MAX_CUBE_MAP_TEXTURE_SIZE: Int\n    val TEXTURE0: Int\n    val TEXTURE1: Int\n    val
TEXTURE2: Int\n    val TEXTURE3: Int\n    val TEXTURE4: Int\n    val TEXTURE5: Int\n    val
TEXTURE6: Int\n    val TEXTURE7: Int\n    val TEXTURE8: Int\n    val TEXTURE9: Int\n    val
TEXTURE10: Int\n    val TEXTURE11: Int\n    val TEXTURE12: Int\n    val TEXTURE13: Int\n    val
TEXTURE14: Int\n    val TEXTURE15: Int\n    val TEXTURE16: Int\n    val TEXTURE17: Int\n    val
TEXTURE18: Int\n    val TEXTURE19: Int\n    val TEXTURE20: Int\n    val TEXTURE21: Int\n    val
TEXTURE22: Int\n    val TEXTURE23: Int\n    val TEXTURE24: Int\n    val TEXTURE25: Int\n    val
TEXTURE26: Int\n    val TEXTURE27: Int\n    val TEXTURE28: Int\n    val TEXTURE29: Int\n    val
TEXTURE30: Int\n    val TEXTURE31: Int\n    val ACTIVE_TEXTURE: Int\n    val REPEAT: Int\n
val CLAMP_TO_EDGE: Int\n    val MIRRORED_REPEAT: Int\n    val FLOAT_VEC2: Int\n    val
FLOAT_VEC3: Int\n    val FLOAT_VEC4: Int\n    val INT_VEC2: Int\n    val INT_VEC3: Int\n    val
INT_VEC4: Int\n    val BOOL: Int\n    val BOOL_VEC2: Int\n    val BOOL_VEC3: Int\n    val
BOOL_VEC4: Int\n    val FLOAT_MAT2: Int\n    val FLOAT_MAT3: Int\n    val FLOAT_MAT4: Int\n
val SAMPLER_2D: Int\n    val SAMPLER_CUBE: Int\n    val VERTEX_ATTRIB_ARRAY_ENABLED:
Int\n    val VERTEX_ATTRIB_ARRAY_SIZE: Int\n    val VERTEX_ATTRIB_ARRAY_STRIDE: Int\n
val VERTEX_ATTRIB_ARRAY_TYPE: Int\n    val VERTEX_ATTRIB_ARRAY_NORMALIZED: Int\n
val VERTEX_ATTRIB_ARRAY_POINTER: Int\n    val VERTEX_ATTRIB_ARRAY_BUFFER_BINDING:
Int\n    val IMPLEMENTATION_COLOR_READ_TYPE: Int\n    val
IMPLEMENTATION_COLOR_READ_FORMAT: Int\n    val COMPILE_STATUS: Int\n    val
LOW_FLOAT: Int\n    val MEDIUM_FLOAT: Int\n    val HIGH_FLOAT: Int\n    val LOW_INT: Int\n
val MEDIUM_INT: Int\n    val HIGH_INT: Int\n    val FRAMEBUFFER: Int\n    val RENDERBUFFER:
Int\n    val RGBA4: Int\n    val RGB5_A1: Int\n    val RGB565: Int\n    val DEPTH_COMPONENT16:
Int\n    val STENCIL_INDEX: Int\n    val STENCIL_INDEX8: Int\n    val DEPTH_STENCIL: Int\n    val
RENDERBUFFER_WIDTH: Int\n    val RENDERBUFFER_HEIGHT: Int\n    val
RENDERBUFFER_INTERNAL_FORMAT: Int\n    val RENDERBUFFER_RED_SIZE: Int\n    val
RENDERBUFFER_GREEN_SIZE: Int\n    val RENDERBUFFER_BLUE_SIZE: Int\n    val
RENDERBUFFER_ALPHA_SIZE: Int\n    val RENDERBUFFER_DEPTH_SIZE: Int\n    val
RENDERBUFFER_STENCIL_SIZE: Int\n    val FRAMEBUFFER_ATTACHMENT_OBJECT_TYPE: Int\n
val FRAMEBUFFER_ATTACHMENT_OBJECT_NAME: Int\n    val
FRAMEBUFFER_ATTACHMENT_TEXTURE_LEVEL: Int\n    val
FRAMEBUFFER_ATTACHMENT_TEXTURE_CUBE_MAP_FACE: Int\n    val COLOR_ATTACHMENT0:
Int\n    val DEPTH_ATTACHMENT: Int\n    val STENCIL_ATTACHMENT: Int\n    val
DEPTH_STENCIL_ATTACHMENT: Int\n    val NONE: Int\n    val FRAMEBUFFER_COMPLETE: Int\n
val FRAMEBUFFER_INCOMPLETE_ATTACHMENT: Int\n    val
FRAMEBUFFER_INCOMPLETE_MISSING_ATTACHMENT: Int\n    val
FRAMEBUFFER_INCOMPLETE_DIMENSIONS: Int\n    val FRAMEBUFFER_UNSUPPORTED: Int\n
val FRAMEBUFFER_BINDING: Int\n    val RENDERBUFFER_BINDING: Int\n    val
MAX_RENDERBUFFER_SIZE: Int\n    val INVALID_FRAMEBUFFER_OPERATION: Int\n    val
UNPACK_FLIP_Y_WEBGL: Int\n    val UNPACK_PREMULTIPLY_ALPHA_WEBGL: Int\n    val
CONTEXT_LOST_WEBGL: Int\n    val UNPACK_COLORSPACE_CONVERSION_WEBGL: Int\n    val
BROWSER_DEFAULT_WEBGL: Int\n    }\n}\n\n/**\n * Exposes the JavaScript
[WebGLRenderingContext](https://developer.mozilla.org/en/docs/Web/API/WebGLRenderingContext) to Kotlin\n
*\npublic external abstract class WebGLRenderingContext : WebGLRenderingContextBase, RenderingContext {\n
companion object {\n    val DEPTH_BUFFER_BIT: Int\n    val STENCIL_BUFFER_BIT: Int\n    val
COLOR_BUFFER_BIT: Int\n    val POINTS: Int\n    val LINES: Int\n    val LINE_LOOP: Int\n    val
LINE_STRIP: Int\n    val TRIANGLES: Int\n    val TRIANGLE_STRIP: Int\n    val TRIANGLE_FAN:

```

Int\n val ZERO: Int\n val ONE: Int\n val SRC\_COLOR: Int\n val ONE\_MINUS\_SRC\_COLOR:  
 Int\n val SRC\_ALPHA: Int\n val ONE\_MINUS\_SRC\_ALPHA: Int\n val DST\_ALPHA: Int\n val  
 ONE\_MINUS\_DST\_ALPHA: Int\n val DST\_COLOR: Int\n val ONE\_MINUS\_DST\_COLOR: Int\n  
 val SRC\_ALPHA\_SATURATE: Int\n val FUNC\_ADD: Int\n val BLEND\_EQUATION: Int\n val  
 BLEND\_EQUATION\_RGB: Int\n val BLEND\_EQUATION\_ALPHA: Int\n val FUNC\_SUBTRACT:  
 Int\n val FUNC\_REVERSE\_SUBTRACT: Int\n val BLEND\_DST\_RGB: Int\n val  
 BLEND\_SRC\_RGB: Int\n val BLEND\_DST\_ALPHA: Int\n val BLEND\_SRC\_ALPHA: Int\n val  
 CONSTANT\_COLOR: Int\n val ONE\_MINUS\_CONSTANT\_COLOR: Int\n val CONSTANT\_ALPHA:  
 Int\n val ONE\_MINUS\_CONSTANT\_ALPHA: Int\n val BLEND\_COLOR: Int\n val  
 ARRAY\_BUFFER: Int\n val ELEMENT\_ARRAY\_BUFFER: Int\n val ARRAY\_BUFFER\_BINDING:  
 Int\n val ELEMENT\_ARRAY\_BUFFER\_BINDING: Int\n val STREAM\_DRAW: Int\n val  
 STATIC\_DRAW: Int\n val DYNAMIC\_DRAW: Int\n val BUFFER\_SIZE: Int\n val  
 BUFFER\_USAGE: Int\n val CURRENT\_VERTEX\_ATTRIB: Int\n val FRONT: Int\n val BACK:  
 Int\n val FRONT\_AND\_BACK: Int\n val CULL\_FACE: Int\n val BLEND: Int\n val DITHER:  
 Int\n val STENCIL\_TEST: Int\n val DEPTH\_TEST: Int\n val SCISSOR\_TEST: Int\n val  
 POLYGON\_OFFSET\_FILL: Int\n val SAMPLE\_ALPHA\_TO\_COVERAGE: Int\n val  
 SAMPLE\_COVERAGE: Int\n val NO\_ERROR: Int\n val INVALID\_ENUM: Int\n val  
 INVALID\_VALUE: Int\n val INVALID\_OPERATION: Int\n val OUT\_OF\_MEMORY: Int\n val CW:  
 Int\n val CCW: Int\n val LINE\_WIDTH: Int\n val ALIASED\_POINT\_SIZE\_RANGE: Int\n val  
 ALIASED\_LINE\_WIDTH\_RANGE: Int\n val CULL\_FACE\_MODE: Int\n val FRONT\_FACE: Int\n  
 val DEPTH\_RANGE: Int\n val DEPTH\_WRITEMASK: Int\n val DEPTH\_CLEAR\_VALUE: Int\n val  
 DEPTH\_FUNC: Int\n val STENCIL\_CLEAR\_VALUE: Int\n val STENCIL\_FUNC: Int\n val  
 STENCIL\_FAIL: Int\n val STENCIL\_PASS\_DEPTH\_FAIL: Int\n val STENCIL\_PASS\_DEPTH\_PASS:  
 Int\n val STENCIL\_REF: Int\n val STENCIL\_VALUE\_MASK: Int\n val STENCIL\_WRITEMASK:  
 Int\n val STENCIL\_BACK\_FUNC: Int\n val STENCIL\_BACK\_FAIL: Int\n val  
 STENCIL\_BACK\_PASS\_DEPTH\_FAIL: Int\n val STENCIL\_BACK\_PASS\_DEPTH\_PASS: Int\n val  
 STENCIL\_BACK\_REF: Int\n val STENCIL\_BACK\_VALUE\_MASK: Int\n val  
 STENCIL\_BACK\_WRITEMASK: Int\n val VIEWPORT: Int\n val SCISSOR\_BOX: Int\n val  
 COLOR\_CLEAR\_VALUE: Int\n val COLOR\_WRITEMASK: Int\n val UNPACK\_ALIGNMENT: Int\n  
 val PACK\_ALIGNMENT: Int\n val MAX\_TEXTURE\_SIZE: Int\n val MAX\_VIEWPORT\_DIMS: Int\n  
 val SUBPIXEL\_BITS: Int\n val RED\_BITS: Int\n val GREEN\_BITS: Int\n val BLUE\_BITS: Int\n  
 val ALPHA\_BITS: Int\n val DEPTH\_BITS: Int\n val STENCIL\_BITS: Int\n val  
 POLYGON\_OFFSET\_UNITS: Int\n val POLYGON\_OFFSET\_FACTOR: Int\n val  
 TEXTURE\_BINDING\_2D: Int\n val SAMPLE\_BUFFERS: Int\n val SAMPLES: Int\n val  
 SAMPLE\_COVERAGE\_VALUE: Int\n val SAMPLE\_COVERAGE\_INVERT: Int\n val  
 COMPRESSED\_TEXTURE\_FORMATS: Int\n val DONT\_CARE: Int\n val FASTEST: Int\n val  
 NICEST: Int\n val GENERATE\_MIPMAP\_HINT: Int\n val BYTE: Int\n val UNSIGNED\_BYTE:  
 Int\n val SHORT: Int\n val UNSIGNED\_SHORT: Int\n val INT: Int\n val UNSIGNED\_INT: Int\n  
 val FLOAT: Int\n val DEPTH\_COMPONENT: Int\n val ALPHA: Int\n val RGB: Int\n val  
 RGBA: Int\n val LUMINANCE: Int\n val LUMINANCE\_ALPHA: Int\n val  
 UNSIGNED\_SHORT\_4\_4\_4\_4: Int\n val UNSIGNED\_SHORT\_5\_5\_5\_1: Int\n val  
 UNSIGNED\_SHORT\_5\_6\_5: Int\n val FRAGMENT\_SHADER: Int\n val VERTEX\_SHADER: Int\n  
 val MAX\_VERTEX\_ATTRIBS: Int\n val MAX\_VERTEX\_UNIFORM\_VECTORS: Int\n val  
 MAX\_VARYING\_VECTORS: Int\n val MAX\_COMBINED\_TEXTURE\_IMAGE\_UNITS: Int\n val  
 MAX\_VERTEX\_TEXTURE\_IMAGE\_UNITS: Int\n val MAX\_TEXTURE\_IMAGE\_UNITS: Int\n val  
 MAX\_FRAGMENT\_UNIFORM\_VECTORS: Int\n val SHADER\_TYPE: Int\n val DELETE\_STATUS:  
 Int\n val LINK\_STATUS: Int\n val VALIDATE\_STATUS: Int\n val ATTACHED\_SHADERS: Int\n  
 val ACTIVE\_UNIFORMS: Int\n val ACTIVE\_ATTRIBUTES: Int\n val

SHADING\_LANGUAGE\_VERSION: Int\n val CURRENT\_PROGRAM: Int\n val NEVER: Int\n val  
 LESS: Int\n val EQUAL: Int\n val LEQUAL: Int\n val GREATER: Int\n val NOTEQUAL: Int\n  
 val GEQUAL: Int\n val ALWAYS: Int\n val KEEP: Int\n val REPLACE: Int\n val INCR: Int\n  
 val DECR: Int\n val INVERT: Int\n val INCR\_WRAP: Int\n val DECR\_WRAP: Int\n val  
 VENDOR: Int\n val RENDERER: Int\n val VERSION: Int\n val NEAREST: Int\n val LINEAR:  
 Int\n val NEAREST\_MIPMAP\_NEAREST: Int\n val LINEAR\_MIPMAP\_NEAREST: Int\n val  
 NEAREST\_MIPMAP\_LINEAR: Int\n val LINEAR\_MIPMAP\_LINEAR: Int\n val  
 TEXTURE\_MAG\_FILTER: Int\n val TEXTURE\_MIN\_FILTER: Int\n val TEXTURE\_WRAP\_S: Int\n  
 val TEXTURE\_WRAP\_T: Int\n val TEXTURE\_2D: Int\n val TEXTURE: Int\n val  
 TEXTURE\_CUBE\_MAP: Int\n val TEXTURE\_BINDING\_CUBE\_MAP: Int\n val  
 TEXTURE\_CUBE\_MAP\_POSITIVE\_X: Int\n val TEXTURE\_CUBE\_MAP\_NEGATIVE\_X: Int\n val  
 TEXTURE\_CUBE\_MAP\_POSITIVE\_Y: Int\n val TEXTURE\_CUBE\_MAP\_NEGATIVE\_Y: Int\n val  
 TEXTURE\_CUBE\_MAP\_POSITIVE\_Z: Int\n val TEXTURE\_CUBE\_MAP\_NEGATIVE\_Z: Int\n val  
 MAX\_CUBE\_MAP\_TEXTURE\_SIZE: Int\n val TEXTURE0: Int\n val TEXTURE1: Int\n val  
 TEXTURE2: Int\n val TEXTURE3: Int\n val TEXTURE4: Int\n val TEXTURE5: Int\n val  
 TEXTURE6: Int\n val TEXTURE7: Int\n val TEXTURE8: Int\n val TEXTURE9: Int\n val  
 TEXTURE10: Int\n val TEXTURE11: Int\n val TEXTURE12: Int\n val TEXTURE13: Int\n val  
 TEXTURE14: Int\n val TEXTURE15: Int\n val TEXTURE16: Int\n val TEXTURE17: Int\n val  
 TEXTURE18: Int\n val TEXTURE19: Int\n val TEXTURE20: Int\n val TEXTURE21: Int\n val  
 TEXTURE22: Int\n val TEXTURE23: Int\n val TEXTURE24: Int\n val TEXTURE25: Int\n val  
 TEXTURE26: Int\n val TEXTURE27: Int\n val TEXTURE28: Int\n val TEXTURE29: Int\n val  
 TEXTURE30: Int\n val TEXTURE31: Int\n val ACTIVE\_TEXTURE: Int\n val REPEAT: Int\n  
 val CLAMP\_TO\_EDGE: Int\n val MIRRORED\_REPEAT: Int\n val FLOAT\_VEC2: Int\n val  
 FLOAT\_VEC3: Int\n val FLOAT\_VEC4: Int\n val INT\_VEC2: Int\n val INT\_VEC3: Int\n val  
 INT\_VEC4: Int\n val BOOL: Int\n val BOOL\_VEC2: Int\n val BOOL\_VEC3: Int\n val  
 BOOL\_VEC4: Int\n val FLOAT\_MAT2: Int\n val FLOAT\_MAT3: Int\n val FLOAT\_MAT4: Int\n  
 val SAMPLER\_2D: Int\n val SAMPLER\_CUBE: Int\n val VERTEX\_ATTRIB\_ARRAY\_ENABLED:  
 Int\n val VERTEX\_ATTRIB\_ARRAY\_SIZE: Int\n val VERTEX\_ATTRIB\_ARRAY\_STRIDE: Int\n  
 val VERTEX\_ATTRIB\_ARRAY\_TYPE: Int\n val VERTEX\_ATTRIB\_ARRAY\_NORMALIZED: Int\n  
 val VERTEX\_ATTRIB\_ARRAY\_POINTER: Int\n val VERTEX\_ATTRIB\_ARRAY\_BUFFER\_BINDING:  
 Int\n val IMPLEMENTATION\_COLOR\_READ\_TYPE: Int\n val  
 IMPLEMENTATION\_COLOR\_READ\_FORMAT: Int\n val COMPILE\_STATUS: Int\n val  
 LOW\_FLOAT: Int\n val MEDIUM\_FLOAT: Int\n val HIGH\_FLOAT: Int\n val LOW\_INT: Int\n  
 val MEDIUM\_INT: Int\n val HIGH\_INT: Int\n val FRAMEBUFFER: Int\n val RENDERBUFFER:  
 Int\n val RGBA4: Int\n val RGB5\_A1: Int\n val RGB565: Int\n val DEPTH\_COMPONENT16:  
 Int\n val STENCIL\_INDEX: Int\n val STENCIL\_INDEX8: Int\n val DEPTH\_STENCIL: Int\n val  
 RENDERBUFFER\_WIDTH: Int\n val RENDERBUFFER\_HEIGHT: Int\n val  
 RENDERBUFFER\_INTERNAL\_FORMAT: Int\n val RENDERBUFFER\_RED\_SIZE: Int\n val  
 RENDERBUFFER\_GREEN\_SIZE: Int\n val RENDERBUFFER\_BLUE\_SIZE: Int\n val  
 RENDERBUFFER\_ALPHA\_SIZE: Int\n val RENDERBUFFER\_DEPTH\_SIZE: Int\n val  
 RENDERBUFFER\_STENCIL\_SIZE: Int\n val FRAMEBUFFER\_ATTACHMENT\_OBJECT\_TYPE: Int\n  
 val FRAMEBUFFER\_ATTACHMENT\_OBJECT\_NAME: Int\n val  
 FRAMEBUFFER\_ATTACHMENT\_TEXTURE\_LEVEL: Int\n val  
 FRAMEBUFFER\_ATTACHMENT\_TEXTURE\_CUBE\_MAP\_FACE: Int\n val COLOR\_ATTACHMENT0:  
 Int\n val DEPTH\_ATTACHMENT: Int\n val STENCIL\_ATTACHMENT: Int\n val  
 DEPTH\_STENCIL\_ATTACHMENT: Int\n val NONE: Int\n val FRAMEBUFFER\_COMPLETE: Int\n  
 val FRAMEBUFFER\_INCOMPLETE\_ATTACHMENT: Int\n val  
 FRAMEBUFFER\_INCOMPLETE\_MISSING\_ATTACHMENT: Int\n val

```

FRAMEBUFFER_INCOMPLETE_DIMENSIONS: Int\n    val FRAMEBUFFER_UNSUPPORTED: Int\n
val FRAMEBUFFER_BINDING: Int\n    val RENDERBUFFER_BINDING: Int\n    val
MAX_RENDERBUFFER_SIZE: Int\n    val INVALID_FRAMEBUFFER_OPERATION: Int\n    val
UNPACK_FLIP_Y_WEBGL: Int\n    val UNPACK_PREMULTIPLY_ALPHA_WEBGL: Int\n    val
CONTEXT_LOST_WEBGL: Int\n    val UNPACK_COLORSPACE_CONVERSION_WEBGL: Int\n    val
BROWSER_DEFAULT_WEBGL: Int\n    }\n}\n\n/**\n * Exposes the JavaScript
[WebGLContextEvent](https://developer.mozilla.org/en/docs/Web/API/WebGLContextEvent) to Kotlin\n
*/\n\npublic external open class WebGLContextEvent(type: String, eventInit: WebGLContextEventInit =
definedExternally) : Event {\n    open val statusMessage: String\n\n    companion object {\n        val NONE: Short\n        val CAPTURING_PHASE: Short\n        val AT_TARGET: Short\n        val BUBBLING_PHASE: Short\n    }\n}\n\npublic external interface WebGLContextEventInit : EventInit {\n    var statusMessage: String? /* = "" */\n    get() = definedExternally\n    set(value) = definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun
WebGLContextEventInit(statusMessage: String? = "", bubbles: Boolean? = false, cancelable: Boolean? = false,
composed: Boolean? = false): WebGLContextEventInit {\n    val o = js("{}")\n    o["statusMessage"] =
statusMessage\n    o["bubbles"] = bubbles\n    o["cancelable"] = cancelable\n    o["composed"] = composed\n
return o\n}\n\n/**\n * Exposes the JavaScript
[ArrayBuffer](https://developer.mozilla.org/en/docs/Web/API/ArrayBuffer) to Kotlin\n
*/\n\npublic external open
class ArrayBuffer(length: Int) : BufferDataSource {\n    open val byteLength: Int\n    fun slice(begin: Int, end: Int =
definedExternally): ArrayBuffer\n\n    companion object {\n        fun isView(value: Any?): Boolean\n    }\n}\n\n/**\n * Exposes the JavaScript
[ArrayBufferView](https://developer.mozilla.org/en/docs/Web/API/ArrayBufferView) to Kotlin\n
*/\n\npublic
external interface ArrayBufferView : BufferDataSource {\n    val buffer: ArrayBuffer\n    val byteOffset: Int\n    val
byteLength: Int\n}\n\n/**\n * Exposes the JavaScript
[Int8Array](https://developer.mozilla.org/en/docs/Web/API/Int8Array) to Kotlin\n
*/\n\npublic external open class
Int8Array : ArrayBufferView {\n    constructor(length: Int)\n    constructor(array: Int8Array)\n    constructor(array:
Array<Byte>)\n    constructor(buffer: ArrayBuffer, byteOffset: Int = definedExternally, length: Int =
definedExternally)\n    open val length: Int\n    override val buffer: ArrayBuffer\n    override val byteOffset: Int\n
override val byteLength: Int\n    fun set(array: Int8Array, offset: Int = definedExternally)\n    fun set(array:
Array<Byte>, offset: Int = definedExternally)\n    fun subarray(start: Int, end: Int): Int8Array\n\n    companion
object {\n        val BYTES_PER_ELEMENT: Int\n    }\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun Int8Array.get(index: Int):
Byte = asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun Int8Array.set(index: Int,
value: Byte) { asDynamic()[index] = value }\n\n/**\n * Exposes the JavaScript
[Uint8Array](https://developer.mozilla.org/en/docs/Web/API/Uint8Array) to Kotlin\n
*/\n\npublic external open class
Uint8Array : ArrayBufferView {\n    constructor(length: Int)\n    constructor(array: Uint8Array)\n
constructor(array: Array<Byte>)\n    constructor(buffer: ArrayBuffer, byteOffset: Int = definedExternally, length:
Int = definedExternally)\n    open val length: Int\n    override val buffer: ArrayBuffer\n    override val byteOffset:
Int\n    override val byteLength: Int\n    fun set(array: Uint8Array, offset: Int = definedExternally)\n    fun set(array:
Array<Byte>, offset: Int = definedExternally)\n    fun subarray(start: Int, end: Int): Uint8Array\n\n    companion
object {\n        val BYTES_PER_ELEMENT: Int\n    }\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun Uint8Array.get(index: Int):
Byte = asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun Uint8Array.set(index: Int,
value: Byte) { asDynamic()[index] = value }\n\n/**\n * Exposes the JavaScript
[Uint8ClampedArray](https://developer.mozilla.org/en/docs/Web/API/Uint8ClampedArray) to Kotlin\n
*/\n\npublic
external open class Uint8ClampedArray : ArrayBufferView {\n    constructor(length: Int)\n    constructor(array:

```

```

    Uint8ClampedArray)\n    constructor(array: Array<Byte>)\n    constructor(buffer: ArrayBuffer, byteOffset: Int =
    definedExternally, length: Int = definedExternally)\n    open val length: Int\n    override val buffer: ArrayBuffer\n    override val byteOffset: Int\n    override val byteLength: Int\n    fun set(array: Uint8ClampedArray, offset: Int =
    definedExternally)\n    fun set(array: Array<Byte>, offset: Int = definedExternally)\n    fun subarray(start: Int, end:
    Int): Uint8ClampedArray\n\n    companion object {\n        val BYTES_PER_ELEMENT: Int\n    }\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun
    Uint8ClampedArray.get(index: Int): Byte = asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun
    Uint8ClampedArray.set(index: Int, value: Byte) { asDynamic()[index] = value }\n\n/**\n * Exposes the JavaScript
    [Int16Array](https://developer.mozilla.org/en/docs/Web/API/Int16Array) to Kotlin\n */\npublic external open class
    Int16Array : ArrayBufferView {\n    constructor(length: Int)\n    constructor(array: Int16Array)\n
    constructor(array: Array<Short>)\n    constructor(buffer: ArrayBuffer, byteOffset: Int = definedExternally, length:
    Int = definedExternally)\n    open val length: Int\n    override val buffer: ArrayBuffer\n    override val byteOffset:
    Int\n    override val byteLength: Int\n    fun set(array: Int16Array, offset: Int = definedExternally)\n    fun set(array:
    Array<Short>, offset: Int = definedExternally)\n    fun subarray(start: Int, end: Int): Int16Array\n\n    companion
    object {\n        val BYTES_PER_ELEMENT: Int\n    }\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun Int16Array.get(index: Int):
    Short = asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun Int16Array.set(index: Int,
    value: Short) { asDynamic()[index] = value }\n\n/**\n * Exposes the JavaScript
    [Uint16Array](https://developer.mozilla.org/en/docs/Web/API/Uint16Array) to Kotlin\n */\npublic external open
    class Uint16Array : ArrayBufferView {\n    constructor(length: Int)\n    constructor(array: Uint16Array)\n
    constructor(array: Array<Short>)\n    constructor(buffer: ArrayBuffer, byteOffset: Int = definedExternally, length:
    Int = definedExternally)\n    open val length: Int\n    override val buffer: ArrayBuffer\n    override val byteOffset:
    Int\n    override val byteLength: Int\n    fun set(array: Uint16Array, offset: Int = definedExternally)\n    fun set(array:
    Array<Short>, offset: Int = definedExternally)\n    fun subarray(start: Int, end: Int): Uint16Array\n\n    companion
    object {\n        val BYTES_PER_ELEMENT: Int\n    }\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun Uint16Array.get(index: Int):
    Short = asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun Uint16Array.set(index: Int,
    value: Short) { asDynamic()[index] = value }\n\n/**\n * Exposes the JavaScript
    [Int32Array](https://developer.mozilla.org/en/docs/Web/API/Int32Array) to Kotlin\n */\npublic external open class
    Int32Array : ArrayBufferView {\n    constructor(length: Int)\n    constructor(array: Int32Array)\n
    constructor(array: Array<Int>)\n    constructor(buffer: ArrayBuffer, byteOffset: Int = definedExternally, length: Int
    = definedExternally)\n    open val length: Int\n    override val buffer: ArrayBuffer\n    override val byteOffset: Int\n
    override val byteLength: Int\n    fun set(array: Int32Array, offset: Int = definedExternally)\n    fun set(array:
    Array<Int>, offset: Int = definedExternally)\n    fun subarray(start: Int, end: Int): Int32Array\n\n    companion
    object {\n        val BYTES_PER_ELEMENT: Int\n    }\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun Int32Array.get(index: Int): Int
    = asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun Int32Array.set(index: Int,
    value: Int) { asDynamic()[index] = value }\n\n/**\n * Exposes the JavaScript
    [Uint32Array](https://developer.mozilla.org/en/docs/Web/API/Uint32Array) to Kotlin\n */\npublic external open
    class Uint32Array : ArrayBufferView {\n    constructor(length: Int)\n    constructor(array: Uint32Array)\n
    constructor(array: Array<Int>)\n    constructor(buffer: ArrayBuffer, byteOffset: Int = definedExternally, length: Int
    = definedExternally)\n    open val length: Int\n    override val buffer: ArrayBuffer\n    override val byteOffset: Int\n
    override val byteLength: Int\n    fun set(array: Uint32Array, offset: Int = definedExternally)\n    fun set(array:

```

```

Array<Int>, offset: Int = definedExternally)\n fun subarray(start: Int, end: Int): Uint32Array\n\n companion
object {\n val BYTES_PER_ELEMENT: Int\n }\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun Uint32Array.get(index: Int):
Int = asDynamic()[index]\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun Uint32Array.set(index: Int,
value: Int) { asDynamic()[index] = value }\n\n/**\n * Exposes the JavaScript
[Float32Array](https://developer.mozilla.org/en/docs/Web/API/Float32Array) to Kotlin\n */\npublic external open
class Float32Array : ArrayBufferView {\n constructor(length: Int)\n constructor(array: Float32Array)\n
constructor(array: Array<Float>)\n constructor(buffer: ArrayBuffer, byteOffset: Int = definedExternally, length:
Int = definedExternally)\n open val length: Int\n override val buffer: ArrayBuffer\n override val byteOffset:
Int\n override val byteLength: Int\n fun set(array: Float32Array, offset: Int = definedExternally)\n fun
set(array: Array<Float>, offset: Int = definedExternally)\n fun subarray(start: Int, end: Int): Float32Array\n\n
companion object {\n val BYTES_PER_ELEMENT: Int\n
}\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun Float32Array.get(index: Int):
Float = asDynamic()[index]\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun Float32Array.set(index: Int,
value: Float) { asDynamic()[index] = value }\n\n/**\n * Exposes the JavaScript
[Float64Array](https://developer.mozilla.org/en/docs/Web/API/Float64Array) to Kotlin\n */\npublic external open
class Float64Array : ArrayBufferView {\n constructor(length: Int)\n constructor(array: Float64Array)\n
constructor(array: Array<Double>)\n constructor(buffer: ArrayBuffer, byteOffset: Int = definedExternally, length:
Int = definedExternally)\n open val length: Int\n override val buffer: ArrayBuffer\n override val byteOffset:
Int\n override val byteLength: Int\n fun set(array: Float64Array, offset: Int = definedExternally)\n fun
set(array: Array<Double>, offset: Int = definedExternally)\n fun subarray(start: Int, end: Int): Float64Array\n\n
companion object {\n val BYTES_PER_ELEMENT: Int\n
}\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun Float64Array.get(index: Int):
Double = asDynamic()[index]\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun Float64Array.set(index: Int,
value: Double) { asDynamic()[index] = value }\n\n/**\n * Exposes the JavaScript
[DataView](https://developer.mozilla.org/en/docs/Web/API/DataView) to Kotlin\n */\npublic external open class
DataView(buffer: ArrayBuffer, byteOffset: Int = definedExternally, byteLength: Int = definedExternally) :
ArrayBufferView {\n override val buffer: ArrayBuffer\n override val byteOffset: Int\n override val
byteLength: Int\n fun getInt8(byteOffset: Int): Byte\n fun getUint8(byteOffset: Int): Byte\n fun
getInt16(byteOffset: Int, littleEndian: Boolean = definedExternally): Short\n fun getUint16(byteOffset: Int,
littleEndian: Boolean = definedExternally): Short\n fun getInt32(byteOffset: Int, littleEndian: Boolean =
definedExternally): Int\n fun getUint32(byteOffset: Int, littleEndian: Boolean = definedExternally): Int\n fun
getFloat32(byteOffset: Int, littleEndian: Boolean = definedExternally): Float\n fun getFloat64(byteOffset: Int,
littleEndian: Boolean = definedExternally): Double\n fun setInt8(byteOffset: Int, value: Byte)\n fun
setUint8(byteOffset: Int, value: Byte)\n fun setInt16(byteOffset: Int, value: Short, littleEndian: Boolean =
definedExternally)\n fun setUint16(byteOffset: Int, value: Short, littleEndian: Boolean = definedExternally)\n
fun setInt32(byteOffset: Int, value: Int, littleEndian: Boolean = definedExternally)\n fun setUint32(byteOffset: Int,
value: Int, littleEndian: Boolean = definedExternally)\n fun setFloat32(byteOffset: Int, value: Float, littleEndian:
Boolean = definedExternally)\n fun setFloat64(byteOffset: Int, value: Double, littleEndian: Boolean =
definedExternally)\n}\n\npublic external interface BufferDataSource\n\npublic external interface
TexImageSource\", \"/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n *
Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*/\n\n// NOTE: THIS FILE IS AUTO-GENERATED, DO NOT EDIT!\n\n// See github.com/kotlin/dukat for

```

```

details\n\npackage org.w3c.dom.clipboard\n\nimport kotlin.js.*\nimport org.khronos.webgl.*\nimport
org.w3c.dom.*\nimport org.w3c.dom.events.*\n\npublic external interface ClipboardEventInit : EventInit {\n    var
clipboardData: DataTransfer? /* = null */\n        get() = definedExternally\n        set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun ClipboardEventInit(clipboardData:
DataTransfer? = null, bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false):
ClipboardEventInit {\n    val o = js(\"({})\")\n    o[\"clipboardData\"] = clipboardData\n    o[\"bubbles\"] = bubbles\n    o[\"cancelable\"] = cancelable\n    o[\"composed\"] = composed\n    return o\n}\n\n/**\n * Exposes the JavaScript
[ClipboardEvent](https://developer.mozilla.org/en/docs/Web/API/ClipboardEvent) to Kotlin\n */\n\npublic external
open class ClipboardEvent(type: String, eventInitDict: ClipboardEventInit = definedExternally) : Event {\n    open
val clipboardData: DataTransfer?\n\n    companion object {\n        val NONE: Short\n        val
CAPTURING_PHASE: Short\n        val AT_TARGET: Short\n        val BUBBLING_PHASE: Short\n    }\n}\n\n/**\n * Exposes the JavaScript [Clipboard](https://developer.mozilla.org/en/docs/Web/API/Clipboard) to
Kotlin\n */\n\npublic external abstract class Clipboard : EventTarget {\n    fun read(): Promise<DataTransfer>\n    fun
readText(): Promise<String>\n    fun write(data: DataTransfer): Promise<Unit>\n    fun writeText(data: String):
Promise<Unit>\n}\n\npublic external interface ClipboardPermissionDescriptor {\n    var allowWithoutGesture:
Boolean? /* = false */\n        get() = definedExternally\n        set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun
ClipboardPermissionDescriptor(allowWithoutGesture: Boolean? = false): ClipboardPermissionDescriptor {\n    val
o = js(\"({})\")\n    o[\"allowWithoutGesture\"] = allowWithoutGesture\n    return o\n}, /*\n * Copyright 2010-
2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n// NOTE: THIS FILE IS AUTO-
GENERATED, DO NOT EDIT!\n\n// See github.com/kotlin/dukat for details\n\npackage org.w3c.dom.css\n\nimport
kotlin.js.*\nimport org.khronos.webgl.*\nimport org.w3c.dom.*\n\npublic external abstract class MediaList :
ItemArrayLike<String> {\n    open var mediaText: String\n    fun appendMedium(medium: String)\n    fun
deleteMedium(medium: String)\n    override fun item(index: Int):
String?\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline operator fun MediaList.get(index: Int):
String? = asDynamic()[index]\n\n/**\n * Exposes the JavaScript
[StyleSheet](https://developer.mozilla.org/en/docs/Web/API/StyleSheet) to Kotlin\n */\n\npublic external abstract
class StyleSheet {\n    open val type: String\n    open val href: String?\n    open val ownerNode:
UnionElementOrProcessingInstruction?\n    open val parentStyleSheet: StyleSheet?\n    open val title: String?\n    open
val media: MediaList\n    open var disabled: Boolean\n}\n\n/**\n * Exposes the JavaScript
[CSSStyleSheet](https://developer.mozilla.org/en/docs/Web/API/CSSStyleSheet) to Kotlin\n */\n\npublic external
abstract class CSSStyleSheet : StyleSheet {\n    open val ownerRule: CSSRule?\n    open val cssRules:
CSSRuleList\n    fun insertRule(rule: String, index: Int): Int\n    fun deleteRule(index: Int)\n}\n\n/**\n * Exposes the
JavaScript [StyleSheetList](https://developer.mozilla.org/en/docs/Web/API/StyleSheetList) to Kotlin\n */\n\npublic
external abstract class StyleSheetList : ItemArrayLike<StyleSheet> {\n    override fun item(index: Int):
StyleSheet?\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline operator fun StyleSheetList.get(index: Int):
StyleSheet? = asDynamic()[index]\n\n/**\n * Exposes the JavaScript
[LinkStyle](https://developer.mozilla.org/en/docs/Web/API/LinkStyle) to Kotlin\n */\n\npublic external interface
LinkStyle {\n    val sheet: StyleSheet?\n        get() = definedExternally\n}\n\n/**\n * Exposes the JavaScript
[CSSRuleList](https://developer.mozilla.org/en/docs/Web/API/CSSRuleList) to Kotlin\n */\n\npublic external abstract
class CSSRuleList : ItemArrayLike<CSSRule> {\n    override fun item(index: Int):
CSSRule?\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline operator fun CSSRuleList.get(index: Int):

```

CSSRule? = asDynamic()[index]\n\n/\*\*\n \* Exposes the JavaScript [CSSRule](https://developer.mozilla.org/en/docs/Web/API/CSSRule) to Kotlin\n \*/\n\npublic external abstract class CSSRule {\n open val type: Short\n open var cssText: String\n open val parentRule: CSSRule?\n open val parentStyleSheet: CSSStyleSheet?\n\n companion object {\n val STYLE\_RULE: Short\n val CHARSET\_RULE: Short\n val IMPORT\_RULE: Short\n val MEDIA\_RULE: Short\n val FONT\_FACE\_RULE: Short\n val PAGE\_RULE: Short\n val MARGIN\_RULE: Short\n val NAMESPACE\_RULE: Short\n }\n}\n\n/\*\*\n \* Exposes the JavaScript [CSSStyleRule](https://developer.mozilla.org/en/docs/Web/API/CSSStyleRule) to Kotlin\n \*/\n\npublic external abstract class CSSStyleRule : CSSRule {\n open var selectorText: String\n open val style: CSSStyleDeclaration\n\n companion object {\n val STYLE\_RULE: Short\n val CHARSET\_RULE: Short\n val IMPORT\_RULE: Short\n val MEDIA\_RULE: Short\n val FONT\_FACE\_RULE: Short\n val PAGE\_RULE: Short\n val MARGIN\_RULE: Short\n val NAMESPACE\_RULE: Short\n }\n}\n\npublic external abstract class CSSImportRule : CSSRule {\n open val href: String\n open val media: MediaList\n open val styleSheet: CSSStyleSheet\n\n companion object {\n val STYLE\_RULE: Short\n val CHARSET\_RULE: Short\n val IMPORT\_RULE: Short\n val MEDIA\_RULE: Short\n val FONT\_FACE\_RULE: Short\n val PAGE\_RULE: Short\n val MARGIN\_RULE: Short\n val NAMESPACE\_RULE: Short\n }\n}\n\n/\*\*\n \* Exposes the JavaScript [CSSGroupingRule](https://developer.mozilla.org/en/docs/Web/API/CSSGroupingRule) to Kotlin\n \*/\n\npublic external abstract class CSSGroupingRule : CSSRule {\n open val cssRules: CSSRuleList\n fun insertRule(rule: String, index: Int): Int\n fun deleteRule(index: Int)\n\n companion object {\n val STYLE\_RULE: Short\n val CHARSET\_RULE: Short\n val IMPORT\_RULE: Short\n val MEDIA\_RULE: Short\n val FONT\_FACE\_RULE: Short\n val PAGE\_RULE: Short\n val MARGIN\_RULE: Short\n val NAMESPACE\_RULE: Short\n }\n}\n\n/\*\*\n \* Exposes the JavaScript [CSSMediaRule](https://developer.mozilla.org/en/docs/Web/API/CSSMediaRule) to Kotlin\n \*/\n\npublic external abstract class CSSMediaRule : CSSGroupingRule {\n open val media: MediaList\n\n companion object {\n val STYLE\_RULE: Short\n val CHARSET\_RULE: Short\n val IMPORT\_RULE: Short\n val MEDIA\_RULE: Short\n val FONT\_FACE\_RULE: Short\n val PAGE\_RULE: Short\n val MARGIN\_RULE: Short\n val NAMESPACE\_RULE: Short\n }\n}\n\n/\*\*\n \* Exposes the JavaScript [CSSPageRule](https://developer.mozilla.org/en/docs/Web/API/CSSPageRule) to Kotlin\n \*/\n\npublic external abstract class CSSPageRule : CSSGroupingRule {\n open var selectorText: String\n open val style: CSSStyleDeclaration\n\n companion object {\n val STYLE\_RULE: Short\n val CHARSET\_RULE: Short\n val IMPORT\_RULE: Short\n val MEDIA\_RULE: Short\n val FONT\_FACE\_RULE: Short\n val PAGE\_RULE: Short\n val MARGIN\_RULE: Short\n val NAMESPACE\_RULE: Short\n }\n}\n\npublic external abstract class CSSMarginRule : CSSRule {\n open val name: String\n open val style: CSSStyleDeclaration\n\n companion object {\n val STYLE\_RULE: Short\n val CHARSET\_RULE: Short\n val IMPORT\_RULE: Short\n val MEDIA\_RULE: Short\n val FONT\_FACE\_RULE: Short\n val PAGE\_RULE: Short\n val MARGIN\_RULE: Short\n val NAMESPACE\_RULE: Short\n }\n}\n\n/\*\*\n \* Exposes the JavaScript [CSSNamespaceRule](https://developer.mozilla.org/en/docs/Web/API/CSSNamespaceRule) to Kotlin\n \*/\n\npublic external abstract class CSSNamespaceRule : CSSRule {\n open val namespaceURI: String\n open val prefix: String\n\n companion object {\n val STYLE\_RULE: Short\n val CHARSET\_RULE: Short\n val IMPORT\_RULE: Short\n val MEDIA\_RULE: Short\n val FONT\_FACE\_RULE: Short\n val PAGE\_RULE: Short\n val MARGIN\_RULE: Short\n val NAMESPACE\_RULE: Short\n }\n}\n\n/\*\*\n \* Exposes the JavaScript [CSSStyleDeclaration](https://developer.mozilla.org/en/docs/Web/API/CSSStyleDeclaration) to Kotlin\n \*/\n\npublic external abstract class CSSStyleDeclaration : ItemArrayLike<String> {\n open var cssText: String\n open val parentRule: CSSRule?\n open var cssFloat: String\n open var alignContent: String\n open var alignItems: String\n open var alignSelf: String\n open var animation: String\n open var animationDelay: String\n open

var animationDirection: String\n open var animationDuration: String\n open var animationFillMode: String\n open var animationIterationCount: String\n open var animationName: String\n open var animationPlayState: String\n open var animationTimingFunction: String\n open var backfaceVisibility: String\n open var background: String\n open var backgroundAttachment: String\n open var backgroundClip: String\n open var backgroundColor: String\n open var backgroundImage: String\n open var backgroundOrigin: String\n open var backgroundPosition: String\n open var backgroundRepeat: String\n open var backgroundSize: String\n open var border: String\n open var borderBottom: String\n open var borderBottomColor: String\n open var borderBottomLeftRadius: String\n open var borderBottomRightRadius: String\n open var borderBottomStyle: String\n open var borderBottomWidth: String\n open var borderCollapse: String\n open var borderColor: String\n open var borderImage: String\n open var borderImageOutset: String\n open var borderImageRepeat: String\n open var borderImageSlice: String\n open var borderImageSource: String\n open var borderImageWidth: String\n open var borderLeft: String\n open var borderLeftColor: String\n open var borderLeftStyle: String\n open var borderLeftWidth: String\n open var borderRadius: String\n open var borderRight: String\n open var borderRightColor: String\n open var borderRightStyle: String\n open var borderRightWidth: String\n open var borderSpacing: String\n open var borderStyle: String\n open var borderTop: String\n open var borderTopColor: String\n open var borderTopLeftRadius: String\n open var borderTopRightRadius: String\n open var borderTopStyle: String\n open var borderTopWidth: String\n open var borderWidth: String\n open var bottom: String\n open var boxDecorationBreak: String\n open var boxShadow: String\n open var boxSizing: String\n open var breakAfter: String\n open var breakBefore: String\n open var breakInside: String\n open var captionSide: String\n open var clear: String\n open var clip: String\n open var color: String\n open var columnCount: String\n open var columnFill: String\n open var columnGap: String\n open var columnRule: String\n open var columnRuleColor: String\n open var columnRuleStyle: String\n open var columnRuleWidth: String\n open var columnSpan: String\n open var columnWidth: String\n open var columns: String\n open var content: String\n open var counterIncrement: String\n open var counterReset: String\n open var cursor: String\n open var direction: String\n open var display: String\n open var emptyCells: String\n open var filter: String\n open var flex: String\n open var flexBasis: String\n open var flexDirection: String\n open var flexFlow: String\n open var flexGrow: String\n open var flexShrink: String\n open var flexWrap: String\n open var font: String\n open var fontFamily: String\n open var fontFeatureSettings: String\n open var fontKerning: String\n open var fontLanguageOverride: String\n open var fontSize: String\n open var fontSizeAdjust: String\n open var fontStretch: String\n open var fontStyle: String\n open var fontSynthesis: String\n open var fontVariant: String\n open var fontVariantAlternates: String\n open var fontVariantCaps: String\n open var fontVariantEastAsian: String\n open var fontVariantLigatures: String\n open var fontVariantNumeric: String\n open var fontVariantPosition: String\n open var fontWeight: String\n open var hangingPunctuation: String\n open var height: String\n open var hyphens: String\n open var imageOrientation: String\n open var imageRendering: String\n open var imageResolution: String\n open var imeMode: String\n open var justifyContent: String\n open var left: String\n open var letterSpacing: String\n open var lineBreak: String\n open var lineHeight: String\n open var listStyle: String\n open var listStyleImage: String\n open var listStylePosition: String\n open var listStyleType: String\n open var margin: String\n open var marginBottom: String\n open var marginLeft: String\n open var marginRight: String\n open var marginTop: String\n open var mark: String\n open var markAfter: String\n open var markBefore: String\n open var marks: String\n open var marqueeDirection: String\n open var marqueePlayCount: String\n open var marqueeSpeed: String\n open var marqueeStyle: String\n open var mask: String\n open var maskType: String\n open var maxHeight: String\n open var maxWidth: String\n open var minHeight: String\n open var minWidth: String\n open var navDown: String\n open var navIndex: String\n open var navLeft: String\n open var navRight: String\n open var navUp: String\n open var objectFit: String\n open var objectPosition: String\n open var opacity: String\n open var order: String\n open var orphans: String\n open var outline: String\n open var outlineColor: String\n open var outlineOffset: String\n open var outlineStyle: String\n open var outlineWidth: String\n open var

```

overflowWrap: String\n open var overflowX: String\n open var overflowY: String\n open var padding:
String\n open var paddingBottom: String\n open var paddingLeft: String\n open var paddingRight: String\n
open var paddingTop: String\n open var pageBreakAfter: String\n open var pageBreakBefore: String\n open
var pageBreakInside: String\n open var perspective: String\n open var perspectiveOrigin: String\n open var
phonemes: String\n open var position: String\n open var quotes: String\n open var resize: String\n open var
rest: String\n open var restAfter: String\n open var restBefore: String\n open var right: String\n open var
tabSize: String\n open var tableLayout: String\n open var textAlign: String\n open var textAlignLast: String\n
open var textCombineUpright: String\n open var textDecoration: String\n open var textDecorationColor:
String\n open var textDecorationLine: String\n open var textDecorationStyle: String\n open var textIndent:
String\n open var textJustify: String\n open var textOrientation: String\n open var textOverflow: String\n
open var textShadow: String\n open var textTransform: String\n open var textUnderlinePosition: String\n open
var top: String\n open var transform: String\n open var transformOrigin: String\n open var transformStyle:
String\n open var transition: String\n open var transitionDelay: String\n open var transitionDuration: String\n
open var transitionProperty: String\n open var transitionTimingFunction: String\n open var unicodeBidi:
String\n open var verticalAlign: String\n open var visibility: String\n open var voiceBalance: String\n open
var voiceDuration: String\n open var voicePitch: String\n open var voicePitchRange: String\n open var
voiceRate: String\n open var voiceStress: String\n open var voiceVolume: String\n open var whiteSpace:
String\n open var widows: String\n open var width: String\n open var wordBreak: String\n open var
wordSpacing: String\n open var wordWrap: String\n open var writingMode: String\n open var zIndex: String\n
open var _dashed_attribute: String\n open var _camel_cased_attribute: String\n open var
_webkit_cased_attribute: String\n fun getPropertyValue(property: String): String\n fun
getPropertyPriority(property: String): String\n fun setProperty(property: String, value: String, priority: String =
definedExternally)\n fun setPropertyValue(property: String, value: String)\n fun setPropertyPriority(property:
String, priority: String)\n fun removeProperty(property: String): String\n override fun item(index: Int):
String\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun
CSSStyleDeclaration.get(index: Int): String? = asDynamic()[index]\n\npublic external interface
ElementCSSInlineStyle {\n val style: CSSStyleDeclaration\n}\n\n/**\n * Exposes the JavaScript
[CSS](https://developer.mozilla.org/en/docs/Web/API/CSS) to Kotlin\n *\n\npublic external abstract class CSS {\n
companion object {\n fun escape(ident: String): String\n }\n}\n\npublic external interface
UnionElementOrProcessingInstruction, \"/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n *\n\n// NOTE: THIS FILE IS AUTO-GENERATED, DO NOT EDIT!\n\n// See
github.com/kotlin/dukat for details\n\npackage org.w3c.dom.encryptedmedia\n\nimport kotlin.js.*\nimport
org.khronos.webgl.*\nimport org.w3c.dom.*\nimport org.w3c.dom.events.*\n\n/**\n * Exposes the JavaScript
[MediaKeySystemConfiguration](https://developer.mozilla.org/en/docs/Web/API/MediaKeySystemConfiguration)
to Kotlin\n *\n\npublic external interface MediaKeySystemConfiguration {\n var label: String? /* = \"\" *\n
get() = definedExternally\n set(value) = definedExternally\n var initDataTypes: Array<String>? /* = arrayOf()
*\n
get() = definedExternally\n set(value) = definedExternally\n var audioCapabilities:
Array<MediaKeySystemMediaCapability>? /* = arrayOf() *\n
get() = definedExternally\n set(value) =
definedExternally\n var videoCapabilities: Array<MediaKeySystemMediaCapability>? /* = arrayOf() *\n
get() = definedExternally\n set(value) = definedExternally\n var distinctiveIdentifier:
MediaKeysRequirement? /* = MediaKeysRequirement.OPTIONAL *\n
get() = definedExternally\n
set(value) = definedExternally\n var persistentState: MediaKeysRequirement? /* =
MediaKeysRequirement.OPTIONAL *\n
get() = definedExternally\n set(value) = definedExternally\n
var sessionTypes: Array<String>?\n
get() = definedExternally\n set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun MediaKeySystemConfiguration(label:

```

```

String? = "", initDataTypes: Array<String>? = arrayOf(), audioCapabilities:
Array<MediaKeySystemMediaCapability>? = arrayOf(), videoCapabilities:
Array<MediaKeySystemMediaCapability>? = arrayOf(), distinctiveIdentifier: MediaKeysRequirement? =
MediaKeysRequirement.OPTIONAL, persistentState: MediaKeysRequirement? =
MediaKeysRequirement.OPTIONAL, sessionTypes: Array<String>? = undefined): MediaKeySystemConfiguration
{n val o = js("{}")\n o["label"] = label\n o["initDataTypes"] = initDataTypes\n
o["audioCapabilities"] = audioCapabilities\n o["videoCapabilities"] = videoCapabilities\n
o["distinctiveIdentifier"] = distinctiveIdentifier\n o["persistentState"] = persistentState\n o["sessionTypes"]
= sessionTypes\n return o\n}\n\npublic external interface MediaKeySystemMediaCapability {n var
contentType: String? /* = "" */\n get() = definedExternally\n set(value) = definedExternally\n var
robustness: String? /* = "" */\n get() = definedExternally\n set(value) =
definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun
MediaKeySystemMediaCapability(contentType: String? = "", robustness: String? = ""):
MediaKeySystemMediaCapability {n val o = js("{}")\n o["contentType"] = contentType\n
o["robustness"] = robustness\n return o\n}\n\n/**\n * Exposes the JavaScript
[MediaKeySystemAccess](https://developer.mozilla.org/en/docs/Web/API/MediaKeySystemAccess) to Kotlin\n
*/\n\npublic external abstract class MediaKeySystemAccess {n open val keySystem: String\n fun
getConfiguration(): MediaKeySystemConfiguration\n fun createMediaKeys(): Promise<MediaKeys>\n}\n\n/**\n
* Exposes the JavaScript [MediaKeys](https://developer.mozilla.org/en/docs/Web/API/MediaKeys) to Kotlin\n
*/\n\npublic external abstract class MediaKeys {n fun createSession(sessionType: MediaKeySessionType =
definedExternally): MediaKeySession\n fun setServerCertificate(serverCertificate: dynamic):
Promise<Boolean>\n}\n\n/**\n * Exposes the JavaScript
[MediaKeySession](https://developer.mozilla.org/en/docs/Web/API/MediaKeySession) to Kotlin\n
*/\n\npublic
external abstract class MediaKeySession : EventTarget {n open val sessionId: String\n open val expiration:
Double\n open val closed: Promise<Unit>\n open val keyStatuses: MediaKeyStatusMap\n open var
onkeystatuschange: ((Event) -> dynamic)?\n open var onmessage: ((MessageEvent) -> dynamic)?\n fun
generateRequest(initDataType: String, initData: dynamic): Promise<Unit>\n fun load(sessionId: String):
Promise<Boolean>\n fun update(response: dynamic): Promise<Unit>\n fun close(): Promise<Unit>\n fun
remove(): Promise<Unit>\n}\n\n/**\n * Exposes the JavaScript
[MediaKeyStatusMap](https://developer.mozilla.org/en/docs/Web/API/MediaKeyStatusMap) to Kotlin\n
*/\n\npublic
external abstract class MediaKeyStatusMap {n open val size: Int\n fun has(keyId: dynamic): Boolean\n fun
get(keyId: dynamic): Any?\n}\n\n/**\n * Exposes the JavaScript
[MediaKeyMessageEvent](https://developer.mozilla.org/en/docs/Web/API/MediaKeyMessageEvent) to Kotlin\n
*/\n\npublic external open class MediaKeyMessageEvent(type: String, eventInitDict: MediaKeyMessageEventInit) :
Event {n open val messageType: MediaKeyMessageType\n open val message: ArrayBuffer\n\n companion
object {n val NONE: Short\n val CAPTURING_PHASE: Short\n val AT_TARGET: Short\n val
BUBBLING_PHASE: Short\n }\n}\n\npublic external interface MediaKeyMessageEventInit : EventInit {n var
messageType: MediaKeyMessageType?\n var message:
ArrayBuffer?\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun
MediaKeyMessageEventInit(messageType: MediaKeyMessageType?, message: ArrayBuffer?, bubbles: Boolean? =
false, cancelable: Boolean? = false, composed: Boolean? = false): MediaKeyMessageEventInit {n val o =
js("{}")\n o["messageType"] = messageType\n o["message"] = message\n o["bubbles"] = bubbles\n
o["cancelable"] = cancelable\n o["composed"] = composed\n return o\n}\n\npublic external open class
MediaEncryptedEvent(type: String, eventInitDict: MediaEncryptedEventInit = definedExternally) : Event {n
open val initDataType: String\n open val initData: ArrayBuffer?\n\n companion object {n val NONE:
Short\n val CAPTURING_PHASE: Short\n val AT_TARGET: Short\n val BUBBLING_PHASE:

```

```

Short\n }n}\n\npublic external interface MediaEncryptedEventInit : EventInit {\n  var initDataType: String? /* =
\"\" */\n  get() = definedExternally\n  set(value) = definedExternally\n  var initData: ArrayBuffer? /* = null
*/\n  get() = definedExternally\n  set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun
MediaEncryptedEventInit(initDataType: String? = \"\", initData: ArrayBuffer? = null, bubbles: Boolean? = false,
cancelable: Boolean? = false, composed: Boolean? = false): MediaEncryptedEventInit {\n  val o = js(\"({})\")\n
o[\"initDataType\"] = initData\n  o[\"initData\"] = initData\n  o[\"bubbles\"] = bubbles\n  o[\"cancelable\"]
= cancelable\n  o[\"composed\"] = composed\n  return o\n}\n\n/* please, don't implement this interface!
*/\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\npublic external
interface MediaKeysRequirement {\n  companion object\n}\n\npublic inline val
MediaKeysRequirement.Companion.REQUIRED: MediaKeysRequirement get() =
\"required\".asDynamic().unsafeCast<MediaKeysRequirement>()\n\npublic inline val
MediaKeysRequirement.Companion.OPTIONAL: MediaKeysRequirement get() =
\"optional\".asDynamic().unsafeCast<MediaKeysRequirement>()\n\npublic inline val
MediaKeysRequirement.Companion.NOT_ALLOWED: MediaKeysRequirement get() = \"not-
allowed\".asDynamic().unsafeCast<MediaKeysRequirement>()\n\n/* please, don't implement this interface!
*/\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\npublic external
interface MediaKeySessionType {\n  companion object\n}\n\npublic inline val
MediaKeySessionType.Companion.TEMPORARY: MediaKeySessionType get() =
\"temporary\".asDynamic().unsafeCast<MediaKeySessionType>()\n\npublic inline val
MediaKeySessionType.Companion.PERSISTENT_LICENSE: MediaKeySessionType get() = \"persistent-
license\".asDynamic().unsafeCast<MediaKeySessionType>()\n\n/* please, don't implement this interface!
*/\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\npublic external
interface MediaKeyStatus {\n  companion object\n}\n\npublic inline val MediaKeyStatus.Companion.USABLE:
MediaKeyStatus get() = \"usable\".asDynamic().unsafeCast<MediaKeyStatus>()\n\npublic inline val
MediaKeyStatus.Companion.EXPIRED: MediaKeyStatus get() =
\"expired\".asDynamic().unsafeCast<MediaKeyStatus>()\n\npublic inline val
MediaKeyStatus.Companion.RELEASED: MediaKeyStatus get() =
\"released\".asDynamic().unsafeCast<MediaKeyStatus>()\n\npublic inline val
MediaKeyStatus.Companion.OUTPUT_RESTRICTED: MediaKeyStatus get() = \"output-
restricted\".asDynamic().unsafeCast<MediaKeyStatus>()\n\npublic inline val
MediaKeyStatus.Companion.OUTPUT_DOWNSCALED: MediaKeyStatus get() = \"output-
downscaled\".asDynamic().unsafeCast<MediaKeyStatus>()\n\npublic inline val
MediaKeyStatus.Companion.STATUS_PENDING: MediaKeyStatus get() = \"status-
pending\".asDynamic().unsafeCast<MediaKeyStatus>()\n\npublic inline val
MediaKeyStatus.Companion.INTERNAL_ERROR: MediaKeyStatus get() = \"internal-
error\".asDynamic().unsafeCast<MediaKeyStatus>()\n\n/* please, don't implement this interface!
*/\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\npublic external
interface MediaKeyMessageType {\n  companion object\n}\n\npublic inline val
MediaKeyMessageType.Companion.LICENSE_REQUEST: MediaKeyMessageType get() = \"license-
request\".asDynamic().unsafeCast<MediaKeyMessageType>()\n\npublic inline val
MediaKeyMessageType.Companion.LICENSE_RENEWAL: MediaKeyMessageType get() = \"license-
renewal\".asDynamic().unsafeCast<MediaKeyMessageType>()\n\npublic inline val
MediaKeyMessageType.Companion.LICENSE_RELEASE: MediaKeyMessageType get() = \"license-
release\".asDynamic().unsafeCast<MediaKeyMessageType>()\n\npublic inline val
MediaKeyMessageType.Companion.INDIVIDUALIZATION_REQUEST: MediaKeyMessageType get() =
\"individualization-request\".asDynamic().unsafeCast<MediaKeyMessageType>()\n\n/*\n * Copyright 2010-2021

```

```

JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n// NOTE: THIS FILE IS AUTO-
GENERATED, DO NOT EDIT!\n\n// See github.com/kotlin/dukat for details\n\npackage
org.w3c.dom.events\n\nimport kotlin.js.*\nimport org.khronos.webgl.*\nimport org.w3c.dom.*\n\n/**\n * Exposes
the JavaScript [UIEvent](https://developer.mozilla.org/en/docs/Web/API/UIEvent) to Kotlin\n */\n\npublic external
open class UIEvent(type: String, eventInitDict: UIEventInit = definedExternally) : Event {\n    open val view:
Window?\n    open val detail: Int\n\n    companion object {\n        val NONE: Short\n        val
CAPTURING_PHASE: Short\n        val AT_TARGET: Short\n        val BUBBLING_PHASE: Short\n
    }\n\n    public external interface UIEventInit : EventInit {\n        var view: Window? /* = null */\n        get() =
definedExternally\n        set(value) = definedExternally\n        var detail: Int? /* = 0 */\n        get() =
definedExternally\n        set(value) = definedExternally\n    }\n\n    @Suppress("\nINVISIBLE_REFERENCE",
"\nINVISIBLE_MEMBER")\n    @kotlin.internal.InlineOnly\n    public inline fun UIEventInit(view: Window? = null,
detail: Int? = 0, bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false): UIEventInit
{\n        val o = js("{}")\n        o["view"] = view\n        o["detail"] = detail\n        o["bubbles"] = bubbles\n
        o["cancelable"] = cancelable\n        o["composed"] = composed\n        return o\n    }\n\n    /**\n     * Exposes the JavaScript
[FocusEvent](https://developer.mozilla.org/en/docs/Web/API/FocusEvent) to Kotlin\n     */\n\n    public external open class
FocusEvent(type: String, eventInitDict: FocusEventInit = definedExternally) : UIEvent {\n        open val relatedTarget:
EventTarget?\n\n        companion object {\n            val NONE: Short\n            val CAPTURING_PHASE: Short\n            val
AT_TARGET: Short\n            val BUBBLING_PHASE: Short\n        }\n\n        public external interface FocusEventInit :
UIEventInit {\n            var relatedTarget: EventTarget? /* = null */\n            get() = definedExternally\n            set(value) =
definedExternally\n        }\n\n        @Suppress("\nINVISIBLE_REFERENCE",
"\nINVISIBLE_MEMBER")\n        @kotlin.internal.InlineOnly\n        public inline fun FocusEventInit(relatedTarget:
EventTarget? = null, view: Window? = null, detail: Int? = 0, bubbles: Boolean? = false, cancelable: Boolean? =
false, composed: Boolean? = false): FocusEventInit {\n            val o = js("{}")\n            o["relatedTarget"] =
relatedTarget\n            o["view"] = view\n            o["detail"] = detail\n            o["bubbles"] = bubbles\n            o["cancelable"] =
cancelable\n            o["composed"] = composed\n            return o\n        }\n\n        /**\n         * Exposes the JavaScript
[MouseEvent](https://developer.mozilla.org/en/docs/Web/API/MouseEvent) to Kotlin\n         */\n\n        public external open
class MouseEvent(type: String, eventInitDict: MouseEventInit = definedExternally) : UIEvent,\n        UnionElementOrMouseEvent {\n            open val screenX: Int\n            open val screenY: Int\n            open val clientX: Int\n
            open val clientY: Int\n            open val ctrlKey: Boolean\n            open val shiftKey: Boolean\n            open val altKey: Boolean\n
            open val metaKey: Boolean\n            open val button: Short\n            open val buttons: Short\n            open val relatedTarget:
EventTarget?\n            open val region: String?\n            open val pageX: Double\n            open val pageY: Double\n            open val x:
Double\n            open val y: Double\n            open val offsetX: Double\n            open val offsetY: Double\n            fun
getModifierState(keyArg: String): Boolean\n\n            companion object {\n                val NONE: Short\n                val
CAPTURING_PHASE: Short\n                val AT_TARGET: Short\n                val BUBBLING_PHASE: Short\n            }\n\n            public external interface MouseEventInit : EventModifierInit {\n                var screenX: Int? /* = 0 */\n                get() =
definedExternally\n                set(value) = definedExternally\n                var screenY: Int? /* = 0 */\n                get() =
definedExternally\n                set(value) = definedExternally\n                var clientX: Int? /* = 0 */\n                get() =
definedExternally\n                set(value) = definedExternally\n                var clientY: Int? /* = 0 */\n                get() =
definedExternally\n                set(value) = definedExternally\n                var button: Short? /* = 0 */\n                get() =
definedExternally\n                set(value) = definedExternally\n                var buttons: Short? /* = 0 */\n                get() =
definedExternally\n                set(value) = definedExternally\n                var relatedTarget: EventTarget? /* = null */\n                get()
= definedExternally\n                set(value) = definedExternally\n                var region: String? /* = null */\n                get() =
definedExternally\n                set(value) = definedExternally\n            }\n\n            @Suppress("\nINVISIBLE_REFERENCE",
"\nINVISIBLE_MEMBER")\n            @kotlin.internal.InlineOnly\n            public inline fun MouseEventInit(screenX: Int? = 0,
screenY: Int? = 0, clientX: Int? = 0, clientY: Int? = 0, button: Short? = 0, buttons: Short? = 0, relatedTarget:
EventTarget? = null, region: String? = null, ctrlKey: Boolean? = false, shiftKey: Boolean? = false, altKey: Boolean?
= false, metaKey: Boolean? = false, modifierAltGraph: Boolean? = false, modifierCapsLock: Boolean? = false,

```

```

modifierFn: Boolean? = false, modifierFnLock: Boolean? = false, modifierHyper: Boolean? = false,
modifierNumLock: Boolean? = false, modifierScrollLock: Boolean? = false, modifierSuper: Boolean? = false,
modifierSymbol: Boolean? = false, modifierSymbolLock: Boolean? = false, view: Window? = null, detail: Int? = 0,
bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false): MouseEventInit {\n  val o =
js(\("{})\n  o["screenX"] = screenX\n  o["screenY"] = screenY\n  o["clientX"] = clientX\n  o["clientY"]
= clientY\n  o["button"] = button\n  o["buttons"] = buttons\n  o["relatedTarget"] = relatedTarget\n
o["region"] = region\n  o["ctrlKey"] = ctrlKey\n  o["shiftKey"] = shiftKey\n  o["altKey"] = altKey\n
o["metaKey"] = metaKey\n  o["modifierAltGraph"] = modifierAltGraph\n  o["modifierCapsLock"] =
modifierCapsLock\n  o["modifierFn"] = modifierFn\n  o["modifierFnLock"] = modifierFnLock\n
o["modifierHyper"] = modifierHyper\n  o["modifierNumLock"] = modifierNumLock\n
o["modifierScrollLock"] = modifierScrollLock\n  o["modifierSuper"] = modifierSuper\n
o["modifierSymbol"] = modifierSymbol\n  o["modifierSymbolLock"] = modifierSymbolLock\n  o["view"] =
view\n  o["detail"] = detail\n  o["bubbles"] = bubbles\n  o["cancelable"] = cancelable\n  o["composed"] =
composed\n  return o\n}\n\npublic external interface EventModifierInit : UIEventInit {\n  var ctrlKey: Boolean?
/* = false */\n  get() = definedExternally\n  set(value) = definedExternally\n  var shiftKey: Boolean? /* =
false */\n  get() = definedExternally\n  set(value) = definedExternally\n  var altKey: Boolean? /* = false
*/\n  get() = definedExternally\n  set(value) = definedExternally\n  var metaKey: Boolean? /* = false */\n
get() = definedExternally\n  set(value) = definedExternally\n  var modifierAltGraph: Boolean? /* = false */\n
get() = definedExternally\n  set(value) = definedExternally\n  var modifierCapsLock: Boolean? /* = false
*/\n  get() = definedExternally\n  set(value) = definedExternally\n  var modifierFn: Boolean? /* = false */\n
get() = definedExternally\n  set(value) = definedExternally\n  var modifierFnLock: Boolean? /* = false */\n
get() = definedExternally\n  set(value) = definedExternally\n  var modifierHyper: Boolean? /* = false */\n
get() = definedExternally\n  set(value) = definedExternally\n  var modifierNumLock: Boolean? /* = false */\n
get() = definedExternally\n  set(value) = definedExternally\n  var modifierScrollLock: Boolean? /* = false
*/\n  get() = definedExternally\n  set(value) = definedExternally\n  var modifierSuper: Boolean? /* = false
*/\n  get() = definedExternally\n  set(value) = definedExternally\n  var modifierSymbol: Boolean? /* = false
*/\n  get() = definedExternally\n  set(value) = definedExternally\n  var modifierSymbolLock: Boolean? /* =
false */\n  get() = definedExternally\n  set(value) =
definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun EventModifierInit(ctrlKey: Boolean? =
false, shiftKey: Boolean? = false, altKey: Boolean? = false, metaKey: Boolean? = false, modifierAltGraph:
Boolean? = false, modifierCapsLock: Boolean? = false, modifierFn: Boolean? = false, modifierFnLock: Boolean? =
false, modifierHyper: Boolean? = false, modifierNumLock: Boolean? = false, modifierScrollLock: Boolean? = false,
modifierSuper: Boolean? = false, modifierSymbol: Boolean? = false, modifierSymbolLock: Boolean? = false, view:
Window? = null, detail: Int? = 0, bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? =
false): EventModifierInit {\n  val o = js(\("{})\n  o["ctrlKey"] = ctrlKey\n  o["shiftKey"] = shiftKey\n
o["altKey"] = altKey\n  o["metaKey"] = metaKey\n  o["modifierAltGraph"] = modifierAltGraph\n
o["modifierCapsLock"] = modifierCapsLock\n  o["modifierFn"] = modifierFn\n  o["modifierFnLock"] =
modifierFnLock\n  o["modifierHyper"] = modifierHyper\n  o["modifierNumLock"] = modifierNumLock\n
o["modifierScrollLock"] = modifierScrollLock\n  o["modifierSuper"] = modifierSuper\n
o["modifierSymbol"] = modifierSymbol\n  o["modifierSymbolLock"] = modifierSymbolLock\n  o["view"] =
view\n  o["detail"] = detail\n  o["bubbles"] = bubbles\n  o["cancelable"] = cancelable\n  o["composed"] =
composed\n  return o\n}\n\n/**\n * Exposes the JavaScript
[WheelEvent](https://developer.mozilla.org/en/docs/Web/API/WheelEvent) to Kotlin\n */\npublic external open
class WheelEvent(type: String, eventInitDict: WheelEventInit = definedExternally) : MouseEvent {\n  open val
deltaX: Double\n  open val deltaY: Double\n  open val deltaZ: Double\n  open val deltaMode: Int\n\n  companion object {\n    val DOM_DELTA_PIXEL: Int\n    val DOM_DELTA_LINE: Int\n    val
DOM_DELTA_PAGE: Int\n    val NONE: Short\n    val CAPTURING_PHASE: Short\n    val

```

```

AT_TARGET: Short\n    val BUBBLING_PHASE: Short\n    }\n}\n\npublic external interface WheelEventInit :
MouseEventInit {\n    var deltaX: Double? /* = 0.0 */\n        get() = definedExternally\n        set(value) =
definedExternally\n    var deltaY: Double? /* = 0.0 */\n        get() = definedExternally\n        set(value) =
definedExternally\n    var deltaZ: Double? /* = 0.0 */\n        get() = definedExternally\n        set(value) =
definedExternally\n    var deltaMode: Int? /* = 0 */\n        get() = definedExternally\n        set(value) =
definedExternally\n}\n\n@Suppress(\\"INVISIBLE_REFERENCE\",
\\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun WheelEventInit(deltaX: Double? = 0.0,
deltaY: Double? = 0.0, deltaZ: Double? = 0.0, deltaMode: Int? = 0, screenX: Int? = 0, screenY: Int? = 0, clientX:
Int? = 0, clientY: Int? = 0, button: Short? = 0, buttons: Short? = 0, relatedTarget: EventTarget? = null, region:
String? = null, ctrlKey: Boolean? = false, shiftKey: Boolean? = false, altKey: Boolean? = false, metaKey: Boolean?
= false, modifierAltGraph: Boolean? = false, modifierCapsLock: Boolean? = false, modifierFn: Boolean? = false,
modifierFnLock: Boolean? = false, modifierHyper: Boolean? = false, modifierNumLock: Boolean? = false,
modifierScrollLock: Boolean? = false, modifierSuper: Boolean? = false, modifierSymbol: Boolean? = false,
modifierSymbolLock: Boolean? = false, view: Window? = null, detail: Int? = 0, bubbles: Boolean? = false,
cancelable: Boolean? = false, composed: Boolean? = false): WheelEventInit {\n    val o = js(\\"({})\")\n
o[\"deltaX\"] = deltaX\n    o[\"deltaY\"] = deltaY\n    o[\"deltaZ\"] = deltaZ\n    o[\"deltaMode\"] = deltaMode\n
o[\"screenX\"] = screenX\n    o[\"screenY\"] = screenY\n    o[\"clientX\"] = clientX\n    o[\"clientY\"] = clientY\n
o[\"button\"] = button\n    o[\"buttons\"] = buttons\n    o[\"relatedTarget\"] = relatedTarget\n    o[\"region\"] =
region\n    o[\"ctrlKey\"] = ctrlKey\n    o[\"shiftKey\"] = shiftKey\n    o[\"altKey\"] = altKey\n    o[\"metaKey\"] =
metaKey\n    o[\"modifierAltGraph\"] = modifierAltGraph\n    o[\"modifierCapsLock\"] = modifierCapsLock\n
o[\"modifierFn\"] = modifierFn\n    o[\"modifierFnLock\"] = modifierFnLock\n    o[\"modifierHyper\"] =
modifierHyper\n    o[\"modifierNumLock\"] = modifierNumLock\n    o[\"modifierScrollLock\"] =
modifierScrollLock\n    o[\"modifierSuper\"] = modifierSuper\n    o[\"modifierSymbol\"] = modifierSymbol\n
o[\"modifierSymbolLock\"] = modifierSymbolLock\n    o[\"view\"] = view\n    o[\"detail\"] = detail\n
o[\"bubbles\"] = bubbles\n    o[\"cancelable\"] = cancelable\n    o[\"composed\"] = composed\n    return
o\n}\n\n/**\n * Exposes the JavaScript [InputEvent](https://developer.mozilla.org/en/docs/Web/API/InputEvent) to
Kotlin\n */\npublic external open class InputEvent(type: String, eventInitDict: InputEventInit = definedExternally) :
UIEvent {\n    open val data: String\n    open val isComposing: Boolean\n\n    companion object {\n        val NONE:
Short\n        val CAPTURING_PHASE: Short\n        val AT_TARGET: Short\n        val BUBBLING_PHASE:
Short\n    }\n}\n\npublic external interface InputEventInit : UIEventInit {\n    var data: String? /* = \\"\" */\n
get() = definedExternally\n    set(value) = definedExternally\n    var isComposing: Boolean? /* = false */\n
get() = definedExternally\n    set(value) = definedExternally\n}\n\n@Suppress(\\"INVISIBLE_REFERENCE\",
\\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun InputEventInit(data: String? = \\"\",
isComposing: Boolean? = false, view: Window? = null, detail: Int? = 0, bubbles: Boolean? = false, cancelable:
Boolean? = false, composed: Boolean? = false): InputEventInit {\n    val o = js(\\"({})\")\n    o[\"data\"] = data\n
o[\"isComposing\"] = isComposing\n    o[\"view\"] = view\n    o[\"detail\"] = detail\n    o[\"bubbles\"] = bubbles\n
o[\"cancelable\"] = cancelable\n    o[\"composed\"] = composed\n    return o\n}\n\n/**\n * Exposes the JavaScript
[KeyboardEvent](https://developer.mozilla.org/en/docs/Web/API/KeyboardEvent) to Kotlin\n */\npublic external
open class KeyboardEvent(type: String, eventInitDict: KeyboardEventInit = definedExternally) : UIEvent {\n
    open val key: String\n    open val code: String\n    open val location: Int\n    open val ctrlKey: Boolean\n
    open val shiftKey: Boolean\n    open val altKey: Boolean\n    open val metaKey: Boolean\n    open val repeat: Boolean\n
    open val isComposing: Boolean\n    open val charCode: Int\n    open val keyCode: Int\n    open val which: Int\n
    fun getModifierState(keyArg: String): Boolean\n\n    companion object {\n        val
DOM_KEY_LOCATION_STANDARD: Int\n        val DOM_KEY_LOCATION_LEFT: Int\n        val
DOM_KEY_LOCATION_RIGHT: Int\n        val DOM_KEY_LOCATION_NUMPAD: Int\n        val NONE:
Short\n        val CAPTURING_PHASE: Short\n        val AT_TARGET: Short\n        val BUBBLING_PHASE:
Short\n    }\n}\n\npublic external interface KeyboardEventInit : EventModifierInit {\n    var key: String? /* = \\"\" */
*\n    get() = definedExternally\n    set(value) = definedExternally\n    var code: String? /* = \\"\" */\n
get()

```

```

= definedExternally\n    set(value) = definedExternally\n    var location: Int? /* = 0 */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var repeat: Boolean? /* = false */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var isComposing: Boolean? /* = false */\n    get() =
definedExternally\n    set(value) = definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun KeyboardEventInit(key: String? = \"\",
code: String? = \"\", location: Int? = 0, repeat: Boolean? = false, isComposing: Boolean? = false, ctrlKey: Boolean?
= false, shiftKey: Boolean? = false, altKey: Boolean? = false, metaKey: Boolean? = false, modifierAltGraph:
Boolean? = false, modifierCapsLock: Boolean? = false, modifierFn: Boolean? = false, modifierFnLock: Boolean? =
false, modifierHyper: Boolean? = false, modifierNumLock: Boolean? = false, modifierScrollLock: Boolean? = false,
modifierSuper: Boolean? = false, modifierSymbol: Boolean? = false, modifierSymbolLock: Boolean? = false, view:
Window? = null, detail: Int? = 0, bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? =
false): KeyboardEventInit {\n    val o = js(\"({})\")\n    o[\"key\"] = key\n    o[\"code\"] = code\n    o[\"location\"] =
location\n    o[\"repeat\"] = repeat\n    o[\"isComposing\"] = isComposing\n    o[\"ctrlKey\"] = ctrlKey\n
o[\"shiftKey\"] = shiftKey\n    o[\"altKey\"] = altKey\n    o[\"metaKey\"] = metaKey\n    o[\"modifierAltGraph\"] =
modifierAltGraph\n    o[\"modifierCapsLock\"] = modifierCapsLock\n    o[\"modifierFn\"] = modifierFn\n
o[\"modifierFnLock\"] = modifierFnLock\n    o[\"modifierHyper\"] = modifierHyper\n    o[\"modifierNumLock\"] =
modifierNumLock\n    o[\"modifierScrollLock\"] = modifierScrollLock\n    o[\"modifierSuper\"] = modifierSuper\n
o[\"modifierSymbol\"] = modifierSymbol\n    o[\"modifierSymbolLock\"] = modifierSymbolLock\n    o[\"view\"] =
view\n    o[\"detail\"] = detail\n    o[\"bubbles\"] = bubbles\n    o[\"cancelable\"] = cancelable\n    o[\"composed\"] =
composed\n    return o\n}\n\n/**\n * Exposes the JavaScript
[CompositionEvent](https://developer.mozilla.org/en/docs/Web/API/CompositionEvent) to Kotlin\n */\npublic
external open class CompositionEvent(type: String, eventInitDict: CompositionEventInit = definedExternally) :
UIEvent {\n    open val data: String\n\n    companion object {\n        val NONE: Short\n        val
CAPTURING_PHASE: Short\n        val AT_TARGET: Short\n        val BUBBLING_PHASE: Short\n
}\n}\n\npublic external interface CompositionEventInit : UIEventInit {\n    var data: String? /* = \"\" */\n    get() =
definedExternally\n    set(value) = definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun CompositionEventInit(data: String? =
\", view: Window? = null, detail: Int? = 0, bubbles: Boolean? = false, cancelable: Boolean? = false, composed:
Boolean? = false): CompositionEventInit {\n    val o = js(\"({})\")\n    o[\"data\"] = data\n    o[\"view\"] = view\n
o[\"detail\"] = detail\n    o[\"bubbles\"] = bubbles\n    o[\"cancelable\"] = cancelable\n    o[\"composed\"] =
composed\n    return o\n}\n\n/**\n * Exposes the JavaScript
[Event](https://developer.mozilla.org/en/docs/Web/API/Event) to Kotlin\n */\npublic external open class
Event(type: String, eventInitDict: EventInit = definedExternally) {\n    open val type: String\n    open val target:
EventTarget?\n    open val currentTarget: EventTarget?\n    open val eventPhase: Short\n    open val bubbles:
Boolean\n    open val cancelable: Boolean\n    open val defaultPrevented: Boolean\n    open val composed:
Boolean\n    open val isTrusted: Boolean\n    open val timeStamp: Number\n    fun composedPath():
Array<EventTarget>\n    fun stopPropagation()\n    fun stopImmediatePropagation()\n    fun preventDefault()\n
fun initEvent(type: String, bubbles: Boolean, cancelable: Boolean)\n\n    companion object {\n        val NONE:
Short\n        val CAPTURING_PHASE: Short\n        val AT_TARGET: Short\n        val BUBBLING_PHASE:
Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[EventTarget](https://developer.mozilla.org/en/docs/Web/API/EventTarget) to Kotlin\n */\npublic external abstract
class EventTarget {\n    fun addEventListener(type: String, callback: EventListener?, options: dynamic =
definedExternally)\n    fun addEventListener(type: String, callback: ((Event) -> Unit)?, options: dynamic =
definedExternally)\n    fun removeEventListener(type: String, callback: EventListener?, options: dynamic =
definedExternally)\n    fun removeEventListener(type: String, callback: ((Event) -> Unit)?, options: dynamic =
definedExternally)\n    fun dispatchEvent(event: Event): Boolean\n}\n\n/**\n * Exposes the JavaScript
[EventListener](https://developer.mozilla.org/en/docs/Web/API/EventListener) to Kotlin\n */\npublic external
interface EventListener {\n    fun handleEvent(event: Event)\n}\n\", /*\n * Copyright 2010-2021 JetBrains s.r.o. and

```

```

Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n *^/\n\n// NOTE: THIS FILE IS AUTO-GENERATED, DO NOT
EDIT!\n\n// See github.com/kotlin/dukat for details\n\npackage org.w3c.dom\n\nimport kotlin.js.*\nimport
org.khronos.webgl.*\nimport org.w3c.dom.clipboard.*\nimport org.w3c.dom.css.*\nimport
org.w3c.dom.encryptedmedia.*\nimport org.w3c.dom.events.*\nimport org.w3c.dom.mediacapture.*\nimport
org.w3c.dom.mediasource.*\nimport org.w3c.dom.pointerevents.*\nimport org.w3c.dom.svg.*\nimport
org.w3c.fetch.*\nimport org.w3c.files.*\nimport org.w3c.performance.*\nimport org.w3c.workers.*\nimport
org.w3c.xhr.*\n\npublic external abstract class HTMLAllCollection {\n    open val length: Int\n    fun
item(nameOrIndex: String = definedExternally): UnionElementOrHTMLCollection?\n    fun namedItem(name:
String): UnionElementOrHTMLCollection?}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun
HTMLAllCollection.get(index: Int): Element? =
asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun
HTMLAllCollection.get(name: String): UnionElementOrHTMLCollection? = asDynamic()[name]\n\n/**\n *
Exposes the JavaScript
[HTMLFormControlsCollection](https://developer.mozilla.org/en/docs/Web/API/HTMLFormControlsCollection)
to Kotlin\n *^/\n\npublic external abstract class HTMLFormControlsCollection : HTMLCollection\n\n/**\n * Exposes
the JavaScript [RadioNodeList](https://developer.mozilla.org/en/docs/Web/API/RadioNodeList) to Kotlin\n
*^/\n\npublic external abstract class RadioNodeList : NodeList, UnionElementOrRadioNodeList {\n    open var value:
String\n}\n\n/**\n * Exposes the JavaScript
[HTMLOptionsCollection](https://developer.mozilla.org/en/docs/Web/API/HTMLOptionsCollection) to Kotlin\n
*^/\n\npublic external abstract class HTMLOptionsCollection : HTMLCollection {\n    override var length: Int\n    open
var selectedIndex: Int\n    fun add(element: UnionHTMLOptGroupElementOrHTMLOptionElement, before:
dynamic = definedExternally)\n    fun remove(index: Int)\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun
HTMLOptionsCollection.set(index: Int, option: HTMLOptionElement?) { asDynamic()[index] = option }\n\n/**\n *
Exposes the JavaScript [HTMLInputElement](https://developer.mozilla.org/en/docs/Web/API/HTMLInputElement) to
Kotlin\n *^/\n\npublic external abstract class HTMLInputElement : Element, GlobalEventHandlers,
DocumentAndElementEventHandlers, ElementContentEditable, ElementCSSInlineStyle {\n    open var title:
String\n    open var lang: String\n    open var translate: Boolean\n    open var dir: String\n    open val dataset:
DOMStringMap\n    open var hidden: Boolean\n    open var tabIndex: Int\n    open var accessKey: String\n    open
val accessKeyLabel: String\n    open var draggable: Boolean\n    open val dropzone: DOMTokenList\n    open var
contextMenu: HTMLMenuElement?\n    open var spellcheck: Boolean\n    open var innerText: String\n    open val
offsetParent: Element?\n    open val offsetTop: Int\n    open val offsetLeft: Int\n    open val offsetWidth: Int\n    open
val offsetHeight: Int\n    fun click()\n    fun focus()\n    fun blur()\n    fun forceSpellCheck()\n\n    companion object
{\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n
        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val
ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE:
Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n\n}\n\n/**\n * Exposes the JavaScript
[HTMLUnknownElement](https://developer.mozilla.org/en/docs/Web/API/HTMLUnknownElement) to Kotlin\n
*^/\n\npublic external abstract class HTMLUnknownElement : HTMLInputElement {\n    companion object {\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val

```

```

CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE:
Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    } \n} \n \n /** \n * Exposes the JavaScript
[DOMStringMap](https://developer.mozilla.org/en/docs/Web/API/DOMStringMap) to Kotlin \n * \n public external
abstract class DOMStringMap \n \n @ Suppress(\ "INVISIBLE_REFERENCE \ " ,
\ "INVISIBLE_MEMBER \ ") \n @ kotlin.internal.InlineOnly \n public inline operator fun DOMStringMap.get(name:
String): String? = asDynamic()[name] \n \n @ Suppress(\ "INVISIBLE_REFERENCE \ " ,
\ "INVISIBLE_MEMBER \ ") \n @ kotlin.internal.InlineOnly \n public inline operator fun DOMStringMap.set(name:
String, value: String) { asDynamic()[name] = value } \n \n /** \n * Exposes the JavaScript
[HTMLHtmlElement](https://developer.mozilla.org/en/docs/Web/API/HTMLHtmlElement) to Kotlin \n * \n public
external abstract class HTMLHtmlElement : HTMLElement { \n    open var version: String \n \n    companion object
{ \n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n
        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val
ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE:
Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    } \n} \n \n /** \n * Exposes the JavaScript
[HTMLHeadElement](https://developer.mozilla.org/en/docs/Web/API/HTMLHeadElement) to Kotlin \n * \n public
external abstract class HTMLHeadElement : HTMLElement { \n    companion object { \n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    } \n} \n \n /** \n * Exposes the JavaScript
[HTMLTitleElement](https://developer.mozilla.org/en/docs/Web/API/HTMLTitleElement) to Kotlin \n * \n public
external abstract class HTMLTitleElement : HTMLElement { \n    open var text: String \n \n    companion object { \n
        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    } \n} \n \n /** \n * Exposes the JavaScript
[HTMLBaseElement](https://developer.mozilla.org/en/docs/Web/API/HTMLBaseElement) to Kotlin \n * \n public

```

```

external abstract class HTMLBaseElement : HTMLElement {\n  open var href: String\n  open var target:
String\n\n  companion object {\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n
val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE:
Short\n    val ENTITY_NODE: Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val
COMMENT_NODE: Short\n    val DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n
val DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[HTMLLinkElement](https://developer.mozilla.org/en/docs/Web/API/HTMLLinkElement) to Kotlin\n */\npublic
external abstract class HTMLLinkElement : HTMLElement, LinkStyle {\n  open var href: String\n  open var
crossOrigin: String?\n  open var rel: String\n  open var `as`: RequestDestination\n  open val relList:
DOMTokenList\n  open var media: String\n  open var nonce: String\n  open var hreflang: String\n  open var
type: String\n  open val sizes: DOMTokenList\n  open varreferrerPolicy: String\n  open var charset: String\n
open var rev: String\n  open var target: String\n  open var scope: String\n  open var workerType:
WorkerType\n\n  companion object {\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE:
Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[HTMLMetaElement](https://developer.mozilla.org/en/docs/Web/API/HTMLMetaElement) to Kotlin\n */\npublic
external abstract class HTMLMetaElement : HTMLElement {\n  open var name: String\n  open var httpEquiv:
String\n  open var content: String\n  open var scheme: String\n\n  companion object {\n    val
ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val
CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE:
Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[HTMLStyleElement](https://developer.mozilla.org/en/docs/Web/API/HTMLStyleElement) to Kotlin\n */\npublic
external abstract class HTMLStyleElement : HTMLElement, LinkStyle {\n  open var media: String\n  open var
nonce: String\n  open var type: String\n\n  companion object {\n    val ELEMENT_NODE: Short\n    val
ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n

```

```

    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLBodyElement](https://developer.mozilla.org/en/docs/Web/API/HTMLBodyElement) to Kotlin\n */\npublic
external abstract class HTMLBodyElement : HTMLInputElement, WindowEventHandlers {\n    open var text: String\n
open var link: String\n    open var vLink: String\n    open var aLink: String\n    open var bgColor: String\n    open
var background: String\n\n    companion object {\n        val ELEMENT_NODE: Short\n        val
ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val
ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val
PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLHeadingElement](https://developer.mozilla.org/en/docs/Web/API/HTMLHeadingElement) to Kotlin\n */\npublic
external abstract class HTMLHeadingElement : HTMLInputElement {\n    open var align: String\n\n    companion object {\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val
TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE:
Short\n        val ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val
COMMENT_NODE: Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n
        val DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLParagraphElement](https://developer.mozilla.org/en/docs/Web/API/HTMLParagraphElement) to Kotlin\n */\npublic
external abstract class HTMLParagraphElement : HTMLInputElement {\n    open var align: String\n\n    companion object {\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val
TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE:
Short\n        val ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val
COMMENT_NODE: Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n
        val DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLHRElement](https://developer.mozilla.org/en/docs/Web/API/HTMLHRElement) to Kotlin\n */\npublic
external abstract class HTMLHRElement : HTMLInputElement {\n    open var align: String\n    open var color: String\n
open var noShade: Boolean\n    open var size: String\n    open var width: String\n\n    companion object {\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val

```

DOCUMENT\_POSITION\_IMPLEMENTATION\_SPECIFIC: Short\n }\\n\\n/\*\*\\n \* Exposes the JavaScript [HTMLPreElement](https://developer.mozilla.org/en/docs/Web/API/HTMLPreElement) to Kotlin\\n \*\\npublic external abstract class HTMLPreElement : HTMLInputElement {\\n open var width: Int\\n\\n companion object {\\n val ELEMENT\_NODE: Short\\n val ATTRIBUTE\_NODE: Short\\n val TEXT\_NODE: Short\\n val CDATA\_SECTION\_NODE: Short\\n val ENTITY\_REFERENCE\_NODE: Short\\n val ENTITY\_NODE: Short\\n val PROCESSING\_INSTRUCTION\_NODE: Short\\n val COMMENT\_NODE: Short\\n val DOCUMENT\_NODE: Short\\n val DOCUMENT\_TYPE\_NODE: Short\\n val DOCUMENT\_FRAGMENT\_NODE: Short\\n val NOTATION\_NODE: Short\\n val DOCUMENT\_POSITION\_DISCONNECTED: Short\\n val DOCUMENT\_POSITION\_PRECEDING: Short\\n val DOCUMENT\_POSITION\_FOLLOWING: Short\\n val DOCUMENT\_POSITION\_CONTAINS: Short\\n val DOCUMENT\_POSITION\_CONTAINED\_BY: Short\\n val

DOCUMENT\_POSITION\_IMPLEMENTATION\_SPECIFIC: Short\n }\\n\\n/\*\*\\n \* Exposes the JavaScript [HTMLQuoteElement](https://developer.mozilla.org/en/docs/Web/API/HTMLQuoteElement) to Kotlin\\n \*\\npublic external abstract class HTMLQuoteElement : HTMLInputElement {\\n open var cite: String\\n\\n companion object {\\n val ELEMENT\_NODE: Short\\n val ATTRIBUTE\_NODE: Short\\n val TEXT\_NODE: Short\\n val CDATA\_SECTION\_NODE: Short\\n val ENTITY\_REFERENCE\_NODE: Short\\n val ENTITY\_NODE: Short\\n val PROCESSING\_INSTRUCTION\_NODE: Short\\n val COMMENT\_NODE: Short\\n val DOCUMENT\_NODE: Short\\n val DOCUMENT\_TYPE\_NODE: Short\\n val DOCUMENT\_FRAGMENT\_NODE: Short\\n val NOTATION\_NODE: Short\\n val DOCUMENT\_POSITION\_DISCONNECTED: Short\\n val DOCUMENT\_POSITION\_PRECEDING: Short\\n val DOCUMENT\_POSITION\_FOLLOWING: Short\\n val DOCUMENT\_POSITION\_CONTAINS: Short\\n val DOCUMENT\_POSITION\_CONTAINED\_BY: Short\\n val

DOCUMENT\_POSITION\_IMPLEMENTATION\_SPECIFIC: Short\n }\\n\\n/\*\*\\n \* Exposes the JavaScript [HTMLLOListElement](https://developer.mozilla.org/en/docs/Web/API/HTMLLOListElement) to Kotlin\\n \*\\npublic external abstract class HTMLLOListElement : HTMLInputElement {\\n open var reversed: Boolean\\n open var start: Int\\n open var type: String\\n open var compact: Boolean\\n\\n companion object {\\n val ELEMENT\_NODE: Short\\n val ATTRIBUTE\_NODE: Short\\n val TEXT\_NODE: Short\\n val CDATA\_SECTION\_NODE: Short\\n val ENTITY\_REFERENCE\_NODE: Short\\n val ENTITY\_NODE: Short\\n val PROCESSING\_INSTRUCTION\_NODE: Short\\n val COMMENT\_NODE: Short\\n val DOCUMENT\_NODE: Short\\n val DOCUMENT\_TYPE\_NODE: Short\\n val DOCUMENT\_FRAGMENT\_NODE: Short\\n val NOTATION\_NODE: Short\\n val DOCUMENT\_POSITION\_DISCONNECTED: Short\\n val DOCUMENT\_POSITION\_PRECEDING: Short\\n val DOCUMENT\_POSITION\_FOLLOWING: Short\\n val DOCUMENT\_POSITION\_CONTAINS: Short\\n val DOCUMENT\_POSITION\_CONTAINED\_BY: Short\\n val

DOCUMENT\_POSITION\_IMPLEMENTATION\_SPECIFIC: Short\n }\\n\\n/\*\*\\n \* Exposes the JavaScript [HTMLLUListElement](https://developer.mozilla.org/en/docs/Web/API/HTMLLUListElement) to Kotlin\\n \*\\npublic external abstract class HTMLLUListElement : HTMLInputElement {\\n open var compact: Boolean\\n open var type: String\\n\\n companion object {\\n val ELEMENT\_NODE: Short\\n val ATTRIBUTE\_NODE: Short\\n val TEXT\_NODE: Short\\n val CDATA\_SECTION\_NODE: Short\\n val ENTITY\_REFERENCE\_NODE: Short\\n val ENTITY\_NODE: Short\\n val PROCESSING\_INSTRUCTION\_NODE: Short\\n val COMMENT\_NODE: Short\\n val DOCUMENT\_NODE: Short\\n val DOCUMENT\_TYPE\_NODE: Short\\n val DOCUMENT\_FRAGMENT\_NODE: Short\\n val NOTATION\_NODE: Short\\n val DOCUMENT\_POSITION\_DISCONNECTED: Short\\n val DOCUMENT\_POSITION\_PRECEDING: Short\\n val DOCUMENT\_POSITION\_FOLLOWING: Short\\n val DOCUMENT\_POSITION\_CONTAINS: Short\\n val DOCUMENT\_POSITION\_CONTAINED\_BY: Short\\n val

DOCUMENT\_POSITION\_IMPLEMENTATION\_SPECIFIC: Short\n }\\n\\n/\*\*\\n \* Exposes the JavaScript [HTMLLIElement](https://developer.mozilla.org/en/docs/Web/API/HTMLLIElement) to Kotlin\\n \*\\npublic external abstract class HTMLLIElement : HTMLInputElement {\\n open var value: Int\\n open var type: String\\n\\n



```

    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val
CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE:
Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLTimeElement](https://developer.mozilla.org/en/docs/Web/API/HTMLTimeElement) to Kotlin\n *^npublic
external abstract class HTMLTimeElement : HTMLElement {\n    open var dateTime: String\n\n    companion
object {\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE:
Short\n        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val
ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE:
Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLSpanElement](https://developer.mozilla.org/en/docs/Web/API/HTMLSpanElement) to Kotlin\n *^npublic
external abstract class HTMLSpanElement : HTMLElement {\n    companion object {\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLBRElement](https://developer.mozilla.org/en/docs/Web/API/HTMLBRElement) to Kotlin\n *^npublic
external abstract class HTMLBRElement : HTMLElement {\n    open var clear: String\n\n    companion object {\n
        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLHyperlinkElementUtils](https://developer.mozilla.org/en/docs/Web/API/HTMLHyperlinkElementUtils) to
Kotlin\n *^npublic external interface HTMLHyperlinkElementUtils {\n    var href: String\n    val origin: String\n
var protocol: String\n    var username: String\n    var password: String\n    var host: String\n    var hostname:
String\n    var port: String\n    var pathname: String\n    var search: String\n    var hash: String\n}\n\n/**\n * Exposes
the JavaScript [HTMLModElement](https://developer.mozilla.org/en/docs/Web/API/HTMLModElement) to
Kotlin\n *^npublic external abstract class HTMLModElement : HTMLElement {\n    open var cite: String\n    open

```

```

var dateTime: String\n\n companion object {\n    val ELEMENT_NODE: Short\n    val
ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n } }\n\n/**\n * Exposes the JavaScript
[HTMLPictureElement](https://developer.mozilla.org/en/docs/Web/API/HTMLPictureElement) to Kotlin\n
*\npublic external abstract class HTMLPictureElement : HTMLElement {\n    companion object {\n    val
ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val
CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE:
Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n } }\n\n/**\n * Exposes the JavaScript
[HTMLSourceElement](https://developer.mozilla.org/en/docs/Web/API/HTMLSourceElement) to Kotlin\n
*\npublic external abstract class HTMLSourceElement : HTMLElement {\n    open var src: String\n    open var
type: String\n    open var srcset: String\n    open var sizes: String\n    open var media: String\n\n    companion object
{\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n
    val CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val
ENTITY_NODE: Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE:
Short\n    val DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n } }\n\n/**\n * Exposes the JavaScript
[HTMLImageElement](https://developer.mozilla.org/en/docs/Web/API/HTMLImageElement) to Kotlin\n
*\npublic external abstract class HTMLImageElement : HTMLElement, HTMLOrSVGImageElement,
TexImageSource {\n    open var alt: String\n    open var src: String\n    open var srcset: String\n    open var sizes:
String\n    open var crossOrigin: String?\n    open var useMap: String\n    open var isMap: Boolean\n    open var
width: Int\n    open var height: Int\n    open val naturalWidth: Int\n    open val naturalHeight: Int\n    open val
complete: Boolean\n    open val currentSrc: String\n    open var referrerPolicy: String\n    open var name: String\n
open var lowsrc: String\n    open var align: String\n    open var hspace: Int\n    open var vspace: Int\n    open var
longDesc: String\n    open var border: String\n    open val x: Int\n    open val y: Int\n\n    companion object {\n
    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val
CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE:
Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n

```

```

    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLIFrameElement](https://developer.mozilla.org/en/docs/Web/API/HTMLIFrameElement) to Kotlin\n
*\n\npublic external abstract class HTMLIFrameElement : HTMLInputElement {\n    open var src: String\n    open var
srcdoc: String\n    open var name: String\n    open val sandbox: DOMTokenList\n    open var allowFullscreen:
Boolean\n    open var allowUserMedia: Boolean\n    open var width: String\n    open var height: String\n    open var
referrerPolicy: String\n    open val contentDocument: Document?\n    open val contentWindow: Window?\n    open
var align: String\n    open var scrolling: String\n    open var frameBorder: String\n    open var longDesc: String\n
open var marginHeight: String\n    open var marginWidth: String\n    fun getSVGDocument(): Document?\n\n    companion object {\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val
TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE:
Short\n        val ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val
COMMENT_NODE: Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n        val
DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n        val
DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLEmbedElement](https://developer.mozilla.org/en/docs/Web/API/HTMLEmbedElement) to Kotlin\n
*\n\npublic external abstract class HTMLEmbedElement : HTMLInputElement {\n    open var src: String\n    open var
type: String\n    open var width: String\n    open var height: String\n    open var align: String\n    open var name:
String\n    fun getSVGDocument(): Document?\n\n    companion object {\n        val ELEMENT_NODE: Short\n        val
ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val
ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val
PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n        val
DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n        val
DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLObjectElement](https://developer.mozilla.org/en/docs/Web/API/HTMLObjectElement) to Kotlin\n
*\n\npublic external abstract class HTMLObjectElement : HTMLInputElement {\n    open var data: String\n    open var
type: String\n    open var typeMustMatch: Boolean\n    open var name: String\n    open var useMap: String\n    open
val form: HTMLFormElement?\n    open var width: String\n    open var height: String\n    open val
contentDocument: Document?\n    open val contentWindow: Window?\n    open val willValidate: Boolean\n    open
val validity: ValidityState\n    open val validationMessage: String\n    open var align: String\n    open var archive:
String\n    open var code: String\n    open var declare: Boolean\n    open var hspace: Int\n    open var standby:
String\n    open var vspace: Int\n    open var codeBase: String\n    open var codeType: String\n    open var border:
String\n    fun getSVGDocument(): Document?\n    fun checkValidity(): Boolean\n    fun reportValidity():
Boolean\n    fun setCustomValidity(error: String)\n\n    companion object {\n        val ELEMENT_NODE: Short\n       
val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val
ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val
PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n        val
DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n

```

```

    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLParamElement](https://developer.mozilla.org/en/docs/Web/API/HTMLParamElement) to Kotlin\n
*\npublic external abstract class HTMLParamElement : HTMLElement {\n    open var name: String\n    open var
value: String\n    open var type: String\n    open var valueType: String\n\n    companion object {\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLVideoElement](https://developer.mozilla.org/en/docs/Web/API/HTMLVideoElement) to Kotlin\n
*\npublic external abstract class HTMLVideoElement : HTMLMediaElement, CanvasImageSource, TexImageSource {\n
    open var width: Int\n    open var height: Int\n    open val videoWidth: Int\n    open val videoHeight: Int\n
    open var poster: String\n    open var playsInline: Boolean\n\n    companion object {\n        val NETWORK_EMPTY: Short\n
        val NETWORK_IDLE: Short\n        val NETWORK_LOADING: Short\n        val NETWORK_NO_SOURCE:
Short\n        val HAVE_NOTHING: Short\n        val HAVE_METADATA: Short\n        val
HAVE_CURRENT_DATA: Short\n        val HAVE_FUTURE_DATA: Short\n        val HAVE_ENOUGH_DATA:
Short\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n
        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val
ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE:
Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLAudioElement](https://developer.mozilla.org/en/docs/Web/API/HTMLAudioElement) to Kotlin\n
*\npublic external abstract class HTMLAudioElement : HTMLMediaElement {\n    companion object {\n        val
NETWORK_EMPTY: Short\n        val NETWORK_IDLE: Short\n        val NETWORK_LOADING: Short\n
        val NETWORK_NO_SOURCE: Short\n        val HAVE_NOTHING: Short\n        val HAVE_METADATA:
Short\n        val HAVE_CURRENT_DATA: Short\n        val HAVE_FUTURE_DATA: Short\n        val
HAVE_ENOUGH_DATA: Short\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n
        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE:
Short\n        val ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val
COMMENT_NODE: Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n
        val DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLTrackElement](https://developer.mozilla.org/en/docs/Web/API/HTMLTrackElement) to Kotlin\n
*\npublic external abstract class HTMLTrackElement : HTMLElement {\n    open var kind: String\n    open var src: String\n
    open var srclang: String\n    open var label: String\n    open var default: Boolean\n    open val readyState: Short\n
    open val track: TextTrack\n\n    companion object {\n        val NONE: Short\n        val LOADING: Short\n        val

```

```

LOADED: Short\n    val ERROR: Short\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE:
Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLMediaElement](https://developer.mozilla.org/en/docs/Web/API/HTMLMediaElement) to Kotlin\n
*/\n\npublic external abstract class HTMLMediaElement : HTMLElement {\n    open val error: MediaError?\n    open
var src: String\n    open var srcObject: MediaProvider?\n    open val currentSrc: String\n    open var crossOrigin:
String?\n    open val networkState: Short\n    open var preload: String\n    open val buffered: TimeRanges\n    open
val readyState: Short\n    open val seeking: Boolean\n    open var currentTime: Double\n    open val duration:
Double\n    open val paused: Boolean\n    open var defaultPlaybackRate: Double\n    open var playbackRate:
Double\n    open val played: TimeRanges\n    open val seekable: TimeRanges\n    open val ended: Boolean\n    open
var autoplay: Boolean\n    open var loop: Boolean\n    open var controls: Boolean\n    open var volume: Double\n
    open var muted: Boolean\n    open var defaultMuted: Boolean\n    open val audioTracks: AudioTrackList\n    open
val videoTracks: VideoTrackList\n    open val textTracks: TextTrackList\n    open val mediaKeys: MediaKeys?\n
    open var onencrypted: ((Event) -> dynamic)?\n    open var onwaitingforkey: ((Event) -> dynamic)?\n    fun load()\n
    fun canPlayType(type: String): CanPlayTypeResult\n    fun fastSeek(time: Double)\n    fun getStartDate():
dynamic\n    fun play(): Promise<Unit>\n    fun pause()\n    fun addTextTrack(kind: TextTrackKind, label: String =
definedExternally, language: String = definedExternally): TextTrack\n    fun setMediaKeys(mediaKeys:
MediaKeys?): Promise<Unit>\n\n    companion object {\n        val NETWORK_EMPTY: Short\n        val
NETWORK_IDLE: Short\n        val NETWORK_LOADING: Short\n        val NETWORK_NO_SOURCE: Short\n
        val HAVE_NOTHING: Short\n        val HAVE_METADATA: Short\n        val HAVE_CURRENT_DATA:
Short\n        val HAVE_FUTURE_DATA: Short\n        val HAVE_ENOUGH_DATA: Short\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[MediaError](https://developer.mozilla.org/en/docs/Web/API/MediaError) to Kotlin\n
*/\n\npublic external abstract
class MediaError {\n    open val code: Short\n\n    companion object {\n        val MEDIA_ERR_ABORTED: Short\n
        val MEDIA_ERR_NETWORK: Short\n        val MEDIA_ERR_DECODE: Short\n        val
MEDIA_ERR_SRC_NOT_SUPPORTED: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[AudioTrackList](https://developer.mozilla.org/en/docs/Web/API/AudioTrackList) to Kotlin\n
*/\n\npublic external
abstract class AudioTrackList : EventTarget {\n    open val length: Int\n    open var onchange: ((Event) ->
dynamic)?\n    open var onaddtrack: ((TrackEvent) -> dynamic)?\n    open var onremovetrack: ((TrackEvent) ->
dynamic)?\n    fun getTrackById(id: String): AudioTrack?\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\n\npublic inline operator fun AudioTrackList.get(index:
Int): AudioTrack? = asDynamic()[index]\n\n/**\n * Exposes the JavaScript
[AudioTrack](https://developer.mozilla.org/en/docs/Web/API/AudioTrack) to Kotlin\n
*/\n\npublic external abstract

```

```

class AudioTrack : UnionAudioTrackOrTextTrackOrVideoTrack {\n  open val id: String\n  open val kind:
String\n  open val label: String\n  open val language: String\n  open var enabled: Boolean\n  open val
sourceBuffer: SourceBuffer?\n}\n\n/**\n * Exposes the JavaScript
[VideoTrackList](https://developer.mozilla.org/en/docs/Web/API/VideoTrackList) to Kotlin\n */\npublic external
abstract class VideoTrackList : EventTarget {\n  open val length: Int\n  open val selectedIndex: Int\n  open var
onchange: ((Event) -> dynamic)?\n  open var onaddtrack: ((TrackEvent) -> dynamic)?\n  open var
onremovetrack: ((TrackEvent) -> dynamic)?\n  fun getTrackById(id: String):
VideoTrack?\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun VideoTrackList.get(index:
Int): VideoTrack? = asDynamic()[index]\n\n/**\n * Exposes the JavaScript
[VideoTrack](https://developer.mozilla.org/en/docs/Web/API/VideoTrack) to Kotlin\n */\npublic external abstract
class VideoTrack : UnionAudioTrackOrTextTrackOrVideoTrack {\n  open val id: String\n  open val kind:
String\n  open val label: String\n  open val language: String\n  open var selected: Boolean\n  open val
sourceBuffer: SourceBuffer?\n}\n\npublic external abstract class TextTrackList : EventTarget {\n  open val length:
Int\n  open var onchange: ((Event) -> dynamic)?\n  open var onaddtrack: ((TrackEvent) -> dynamic)?\n  open
var onremovetrack: ((TrackEvent) -> dynamic)?\n  fun getTrackById(id: String):
TextTrack?\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun TextTrackList.get(index: Int):
TextTrack? = asDynamic()[index]\n\n/**\n * Exposes the JavaScript
[TextTrack](https://developer.mozilla.org/en/docs/Web/API/TextTrack) to Kotlin\n */\npublic external abstract
class TextTrack : EventTarget, UnionAudioTrackOrTextTrackOrVideoTrack {\n  open val kind: TextTrackKind\n
open val label: String\n  open val language: String\n  open val id: String\n  open val
inBandMetadataTrackDispatchType: String\n  open var mode: TextTrackMode\n  open val cues:
TextTrackCueList?\n  open val activeCues: TextTrackCueList?\n  open var oncuechange: ((Event) ->
dynamic)?\n  open val sourceBuffer: SourceBuffer?\n  fun addCue(cue: TextTrackCue)\n  fun removeCue(cue:
TextTrackCue)\n}\n\npublic external abstract class TextTrackCueList {\n  open val length: Int\n  fun
getCueById(id: String): TextTrackCue?\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun TextTrackCueList.get(index:
Int): TextTrackCue? = asDynamic()[index]\n\n/**\n * Exposes the JavaScript
[TextTrackCue](https://developer.mozilla.org/en/docs/Web/API/TextTrackCue) to Kotlin\n */\npublic external
abstract class TextTrackCue : EventTarget {\n  open val track: TextTrack?\n  open var id: String\n  open var
startTime: Double\n  open var endTime: Double\n  open var pauseOnExit: Boolean\n  open var onenter: ((Event)
-> dynamic)?\n  open var onexit: ((Event) -> dynamic)?\n}\n\n/**\n * Exposes the JavaScript
[TimeRanges](https://developer.mozilla.org/en/docs/Web/API/TimeRanges) to Kotlin\n */\npublic external abstract
class TimeRanges {\n  open val length: Int\n  fun start(index: Int): Double\n  fun end(index: Int):
Double\n}\n\n/**\n * Exposes the JavaScript
[TrackEvent](https://developer.mozilla.org/en/docs/Web/API/TrackEvent) to Kotlin\n */\npublic external open class
TrackEvent(type: String, eventInitDict: TrackEventInit = definedExternally) : Event {\n  open val track:
UnionAudioTrackOrTextTrackOrVideoTrack?\n\n  companion object {\n    val NONE: Short\n    val
CAPTURING_PHASE: Short\n    val AT_TARGET: Short\n    val BUBBLING_PHASE: Short\n
}\n}\n\npublic external interface TrackEventInit : EventInit {\n  var track:
UnionAudioTrackOrTextTrackOrVideoTrack? /* = null */\n  get() = definedExternally\n  set(value) =
definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun TrackEventInit(track:
UnionAudioTrackOrTextTrackOrVideoTrack? = null, bubbles: Boolean? = false, cancelable: Boolean? = false,
composed: Boolean? = false): TrackEventInit {\n  val o = js("{}")\n  o["track"] = track\n  o["bubbles"] =
bubbles\n  o["cancelable"] = cancelable\n  o["composed"] = composed\n  return o\n}\n\n/**\n * Exposes the
JavaScript [HTMLElement](https://developer.mozilla.org/en/docs/Web/API/HTMLElement) to Kotlin\n

```

```

*/\npublic external abstract class HTMLMapElement : HTMLInputElement {\n  open var name: String\n  open val
areas: HTMLCollection\n\n  companion object {\n    val ELEMENT_NODE: Short\n    val
ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[HTMLAreaElement](https://developer.mozilla.org/en/docs/Web/API/HTMLAreaElement) to Kotlin\n
*/\npublic external abstract class HTMLAreaElement : HTMLInputElement, HTMLHyperlinkElementUtils {\n  open var alt:
String\n  open var coords: String\n  open var shape: String\n  open var target: String\n  open var download:
String\n  open var ping: String\n  open var rel: String\n  open val relList: DOMTokenList\n  open var
referrerPolicy: String\n  open var noHref: Boolean\n\n  companion object {\n    val ELEMENT_NODE:
Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE:
Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[HTMLTableElement](https://developer.mozilla.org/en/docs/Web/API/HTMLTableElement) to Kotlin\n
*/\npublic external abstract class HTMLTableElement : HTMLInputElement {\n  open var caption:
HTMLTableCaptionElement?\n  open var tHead: HTMLTableSectionElement?\n  open var tFoot:
HTMLTableSectionElement?\n  open val tBodies: HTMLCollection\n  open val rows: HTMLCollection\n  open
var align: String\n  open var border: String\n  open var frame: String\n  open var rules: String\n  open var
summary: String\n  open var width: String\n  open var bgColor: String\n  open var cellPadding: String\n  open
var cellSpacing: String\n  fun createCaption(): HTMLTableCaptionElement\n  fun deleteCaption()\n  fun
createTHead(): HTMLTableSectionElement\n  fun deleteTHead()\n  fun createTFoot():
HTMLTableSectionElement\n  fun deleteTFoot()\n  fun createTBody(): HTMLTableSectionElement\n  fun
insertRow(index: Int = definedExternally): HTMLTableRowElement\n  fun deleteRow(index: Int)\n\n  companion object {\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val
TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE:
Short\n    val ENTITY_NODE: Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val
COMMENT_NODE: Short\n    val DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n
    val DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[HTMLTableCaptionElement](https://developer.mozilla.org/en/docs/Web/API/HTMLTableCaptionElement) to
Kotlin\n
*/\npublic external abstract class HTMLTableCaptionElement : HTMLInputElement {\n  open var align:
String\n\n  companion object {\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n
    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE:

```

```

Short\n    val ENTITY_NODE: Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val
COMMENT_NODE: Short\n    val DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n
    val DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLTableColElement](https://developer.mozilla.org/en/docs/Web/API/HTMLTableColElement) to Kotlin\n
*/\npublic external abstract class HTMLTableColElement : HTMLInputElement {\n    open var span: Int\n    open var
align: String\n    open var ch: String\n    open var chOff: String\n    open var vAlign: String\n    open var width:
String\n\n    companion object {\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n
        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE:
Short\n        val ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val
COMMENT_NODE: Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n
        val DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLTableSectionElement](https://developer.mozilla.org/en/docs/Web/API/HTMLTableSectionElement) to
Kotlin\n
*/\npublic external abstract class HTMLTableSectionElement : HTMLInputElement {\n    open val rows:
HTMLCollection\n    open var align: String\n    open var ch: String\n    open var chOff: String\n    open var vAlign:
String\n    fun insertRow(index: Int = definedExternally): HTMLInputElement\n    fun deleteRow(index: Int)\n\n
    companion object {\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val
TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE:
Short\n        val ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val
COMMENT_NODE: Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n
        val DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLTableRowElement](https://developer.mozilla.org/en/docs/Web/API/HTMLTableRowElement) to Kotlin\n
*/\npublic external abstract class HTMLTableRowElement : HTMLInputElement {\n    open val rowIndex: Int\n    open
val sectionRowIndex: Int\n    open val cells: HTMLCollection\n    open var align: String\n    open var ch: String\n
open var chOff: String\n    open var vAlign: String\n    open var bgColor: String\n    fun insertCell(index: Int =
definedExternally): HTMLInputElement\n    fun deleteCell(index: Int)\n\n    companion object {\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLTableCellElement](https://developer.mozilla.org/en/docs/Web/API/HTMLTableCellElement) to Kotlin\n
*/\npublic external abstract class HTMLTableCellElement : HTMLInputElement {\n    open var colSpan: Int\n    open var

```

```

rowSpan: Int\n open var headers: String\n open val cellIndex: Int\n open var scope: String\n open var abbr:
String\n open var align: String\n open var axis: String\n open var height: String\n open var width: String\n
open var ch: String\n open var chOff: String\n open var noWrap: Boolean\n open var vAlign: String\n open
var bgColor: String\n\n companion object {\n val ELEMENT_NODE: Short\n val ATTRIBUTE_NODE:
Short\n val TEXT_NODE: Short\n val CDATA_SECTION_NODE: Short\n val
ENTITY_REFERENCE_NODE: Short\n val ENTITY_NODE: Short\n val
PROCESSING_INSTRUCTION_NODE: Short\n val COMMENT_NODE: Short\n val
DOCUMENT_NODE: Short\n val DOCUMENT_TYPE_NODE: Short\n val
DOCUMENT_FRAGMENT_NODE: Short\n val NOTATION_NODE: Short\n val
DOCUMENT_POSITION_DISCONNECTED: Short\n val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n }
}\n\n\n/**\n * Exposes the JavaScript
[HTMLFormElement](https://developer.mozilla.org/en/docs/Web/API/HTMLFormElement) to Kotlin\n *\npublic
external abstract class HTMLFormElement : HTMLElement {\n open var acceptCharset: String\n open var
action: String\n open var autocomplete: String\n open var enctype: String\n open var encoding: String\n open
var method: String\n open var name: String\n open var noValidate: Boolean\n open var target: String\n open
val elements: HTMLFormControlsCollection\n open val length: Int\n fun submit()\n fun reset()\n fun
checkValidity(): Boolean\n fun reportValidity(): Boolean\n\n companion object {\n val ELEMENT_NODE:
Short\n val ATTRIBUTE_NODE: Short\n val TEXT_NODE: Short\n val CDATA_SECTION_NODE:
Short\n val ENTITY_REFERENCE_NODE: Short\n val ENTITY_NODE: Short\n val
PROCESSING_INSTRUCTION_NODE: Short\n val COMMENT_NODE: Short\n val
DOCUMENT_NODE: Short\n val DOCUMENT_TYPE_NODE: Short\n val
DOCUMENT_FRAGMENT_NODE: Short\n val NOTATION_NODE: Short\n val
DOCUMENT_POSITION_DISCONNECTED: Short\n val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n
}\n}\n\n@Suppress(\n"INVISIBLE_REFERENCE",
\n"INVISIBLE_MEMBER")\n\n@kotlin.internal.InlineOnly\n\npublic inline operator fun
HTMLFormElement.get(index: Int): Element? =
asDynamic()[index]\n\n\n@Suppress(\n"INVISIBLE_REFERENCE",
\n"INVISIBLE_MEMBER")\n\n@kotlin.internal.InlineOnly\n\npublic inline operator fun
HTMLFormElement.get(name: String): UnionElementOrRadioNodeList? = asDynamic()[name]\n\n\n\n/**\n * Exposes
the JavaScript [HTMLLabelElement](https://developer.mozilla.org/en/docs/Web/API/HTMLLabelElement) to
Kotlin\n *\npublic external abstract class HTMLLabelElement : HTMLElement {\n open val form:
HTMLFormElement?\n open var htmlFor: String\n open val control: HTMLElement?\n\n\n companion object
{\n val ELEMENT_NODE: Short\n val ATTRIBUTE_NODE: Short\n val TEXT_NODE: Short\n
val CDATA_SECTION_NODE: Short\n val ENTITY_REFERENCE_NODE: Short\n val
ENTITY_NODE: Short\n val PROCESSING_INSTRUCTION_NODE: Short\n val COMMENT_NODE:
Short\n val DOCUMENT_NODE: Short\n val DOCUMENT_TYPE_NODE: Short\n val
DOCUMENT_FRAGMENT_NODE: Short\n val NOTATION_NODE: Short\n val
DOCUMENT_POSITION_DISCONNECTED: Short\n val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n }
}\n\n\n\n/**\n * Exposes the JavaScript
[HTMLInputElement](https://developer.mozilla.org/en/docs/Web/API/HTMLInputElement) to Kotlin\n *\npublic
external abstract class HTMLInputElement : HTMLElement {\n open var accept: String\n open var alt: String\n

```

```

open var autoComplete: String\n open var autofocus: Boolean\n open var defaultChecked: Boolean\n open var
checked: Boolean\n open var dirName: String\n open var disabled: Boolean\n open val form:
HTMLFormElement?\n open val files: FileList?\n open var formAction: String\n open var formEnctype:
String\n open var formMethod: String\n open var formNoValidate: Boolean\n open var formTarget: String\n
open var height: Int\n open var indeterminate: Boolean\n open var inputMode: String\n open val list:
HTMLElement?\n open var max: String\n open var maxLength: Int\n open var min: String\n open var
minLength: Int\n open var multiple: Boolean\n open var name: String\n open var pattern: String\n open var
placeholder: String\n open var readOnly: Boolean\n open var required: Boolean\n open var size: Int\n open
var src: String\n open var step: String\n open var type: String\n open var defaultValue: String\n open var
value: String\n open var valueAsDate: dynamic\n open var valueAsNumber: Double\n open var width: Int\n
open val willValidate: Boolean\n open val validity: ValidityState\n open val validationMessage: String\n open
val labels: NodeList\n open var selectionStart: Int?\n open var selectionEnd: Int?\n open var
selectionDirection: String?\n open var align: String\n open var useMap: String\n fun stepUp(n: Int =
definedExternally)\n fun stepDown(n: Int = definedExternally)\n fun checkValidity(): Boolean\n fun
reportValidity(): Boolean\n fun setCustomValidity(error: String)\n fun select()\n fun
setRangeText(replacement: String)\n fun setRangeText(replacement: String, start: Int, end: Int, selectionMode:
SelectionMode = definedExternally)\n fun setSelectionRange(start: Int, end: Int, direction: String =
definedExternally)\n\n companion object {\n val ELEMENT_NODE: Short\n val ATTRIBUTE_NODE:
Short\n val TEXT_NODE: Short\n val CDATA_SECTION_NODE: Short\n val
ENTITY_REFERENCE_NODE: Short\n val ENTITY_NODE: Short\n val
PROCESSING_INSTRUCTION_NODE: Short\n val COMMENT_NODE: Short\n val
DOCUMENT_NODE: Short\n val DOCUMENT_TYPE_NODE: Short\n val
DOCUMENT_FRAGMENT_NODE: Short\n val NOTATION_NODE: Short\n val
DOCUMENT_POSITION_DISCONNECTED: Short\n val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n }\n}\n\n/**\n * Exposes the JavaScript
[HTMLButtonElement](https://developer.mozilla.org/en/docs/Web/API/HTMLButtonElement) to Kotlin\n
*\npublic external abstract class HTMLButtonElement : HTMLElement {\n open var autofocus: Boolean\n
open var disabled: Boolean\n open val form: HTMLFormElement?\n open var formAction: String\n open var
formEnctype: String\n open var formMethod: String\n open var formNoValidate: Boolean\n open var
formTarget: String\n open var name: String\n open var type: String\n open var value: String\n open var
menu: HTMLMenuElement?\n open val willValidate: Boolean\n open val validity: ValidityState\n open val
validationMessage: String\n open val labels: NodeList\n fun checkValidity(): Boolean\n fun reportValidity():
Boolean\n fun setCustomValidity(error: String)\n\n companion object {\n val ELEMENT_NODE: Short\n
val ATTRIBUTE_NODE: Short\n val TEXT_NODE: Short\n val CDATA_SECTION_NODE: Short\n
val ENTITY_REFERENCE_NODE: Short\n val ENTITY_NODE: Short\n val
PROCESSING_INSTRUCTION_NODE: Short\n val COMMENT_NODE: Short\n val
DOCUMENT_NODE: Short\n val DOCUMENT_TYPE_NODE: Short\n val
DOCUMENT_FRAGMENT_NODE: Short\n val NOTATION_NODE: Short\n val
DOCUMENT_POSITION_DISCONNECTED: Short\n val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n }\n}\n\n/**\n * Exposes the JavaScript
[HTMLSelectElement](https://developer.mozilla.org/en/docs/Web/API/HTMLSelectElement) to Kotlin\n
*\npublic external abstract class HTMLSelectElement : HTMLElement, ItemArrayLike<Element> {\n open var
autocomplete: String\n open var autofocus: Boolean\n open var disabled: Boolean\n open val form:
HTMLFormElement?\n open var multiple: Boolean\n open var name: String\n open var required: Boolean\n

```

```

open var size: Int\n  open val type: String\n  open val options: HTMLOptionsCollection\n  override var length:
Int\n  open val selectedOptions: HTMLCollection\n  open var selectedIndex: Int\n  open var value: String\n
open val willValidate: Boolean\n  open val validity: ValidityState\n  open val validationMessage: String\n  open
val labels: NodeList\n  fun namedItem(name: String): HTMLOptionElement?\n  fun add(element:
UnionHTMLOptGroupElementOrHTMLOptionElement, before: dynamic = definedExternally)\n  fun
remove(index: Int)\n  fun checkValidity(): Boolean\n  fun reportValidity(): Boolean\n  fun
setCustomValidity(error: String)\n  override fun item(index: Int): Element?\n  companion object {\n  val
ELEMENT_NODE: Short\n  val ATTRIBUTE_NODE: Short\n  val TEXT_NODE: Short\n  val
CDATA_SECTION_NODE: Short\n  val ENTITY_REFERENCE_NODE: Short\n  val ENTITY_NODE:
Short\n  val PROCESSING_INSTRUCTION_NODE: Short\n  val COMMENT_NODE: Short\n  val
DOCUMENT_NODE: Short\n  val DOCUMENT_TYPE_NODE: Short\n  val
DOCUMENT_FRAGMENT_NODE: Short\n  val NOTATION_NODE: Short\n  val
DOCUMENT_POSITION_DISCONNECTED: Short\n  val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n  val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n  val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n
}\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun
HTMLSelectElement.get(index: Int): Element? =
asDynamic()[index]\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun
HTMLSelectElement.set(index: Int, option: HTMLOptionElement?) { asDynamic()[index] = option }\n\n/**\n *
Exposes the JavaScript
[HTMLDataListElement](https://developer.mozilla.org/en/docs/Web/API/HTMLDataListElement) to Kotlin\n
*\n\npublic external abstract class HTMLDataListElement : HTMLInputElement {\n  open val options:
HTMLCollection\n\n  companion object {\n  val ELEMENT_NODE: Short\n  val ATTRIBUTE_NODE:
Short\n  val TEXT_NODE: Short\n  val CDATA_SECTION_NODE: Short\n  val
ENTITY_REFERENCE_NODE: Short\n  val ENTITY_NODE: Short\n  val
PROCESSING_INSTRUCTION_NODE: Short\n  val COMMENT_NODE: Short\n  val
DOCUMENT_NODE: Short\n  val DOCUMENT_TYPE_NODE: Short\n  val
DOCUMENT_FRAGMENT_NODE: Short\n  val NOTATION_NODE: Short\n  val
DOCUMENT_POSITION_DISCONNECTED: Short\n  val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n  val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n  val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[HTMLOptGroupElement](https://developer.mozilla.org/en/docs/Web/API/HTMLOptGroupElement) to Kotlin\n
*\n\npublic external abstract class HTMLOptGroupElement : HTMLInputElement,
UnionHTMLOptGroupElementOrHTMLOptionElement {\n  open var disabled: Boolean\n  open var label:
String\n\n  companion object {\n  val ELEMENT_NODE: Short\n  val ATTRIBUTE_NODE: Short\n
val TEXT_NODE: Short\n  val CDATA_SECTION_NODE: Short\n  val ENTITY_REFERENCE_NODE:
Short\n  val ENTITY_NODE: Short\n  val PROCESSING_INSTRUCTION_NODE: Short\n  val
COMMENT_NODE: Short\n  val DOCUMENT_NODE: Short\n  val DOCUMENT_TYPE_NODE: Short\n
val DOCUMENT_FRAGMENT_NODE: Short\n  val NOTATION_NODE: Short\n  val
DOCUMENT_POSITION_DISCONNECTED: Short\n  val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n  val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n  val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[HTMLOptionElement](https://developer.mozilla.org/en/docs/Web/API/HTMLOptionElement) to Kotlin\n

```

```

*\npublic external abstract class HTMLInputElement : HTMLInputElement,
UnionHTMLOptGroupElementOrHTMLInputElement {\n open var disabled: Boolean\n open val form:
HTMLFormElement?\n open var label: String\n open var defaultSelected: Boolean\n open var selected:
Boolean\n open var value: String\n open var text: String\n open val index: Int\n\n companion object {\n
val ELEMENT_NODE: Short\n val ATTRIBUTE_NODE: Short\n val TEXT_NODE: Short\n val
CDATA_SECTION_NODE: Short\n val ENTITY_REFERENCE_NODE: Short\n val ENTITY_NODE:
Short\n val PROCESSING_INSTRUCTION_NODE: Short\n val COMMENT_NODE: Short\n val
DOCUMENT_NODE: Short\n val DOCUMENT_TYPE_NODE: Short\n val
DOCUMENT_FRAGMENT_NODE: Short\n val NOTATION_NODE: Short\n val
DOCUMENT_POSITION_DISCONNECTED: Short\n val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n }\n}\n\n/**\n * Exposes the JavaScript
[HTMLTextAreaElement](https://developer.mozilla.org/en/docs/Web/API/HTMLTextAreaElement) to Kotlin\n
*\npublic external abstract class HTMLTextAreaElement : HTMLInputElement {\n open var autocomplete: String\n
open var autofocus: Boolean\n open var cols: Int\n open var dirName: String\n open var disabled: Boolean\n
open val form: HTMLFormElement?\n open var inputMode: String\n open var maxLength: Int\n open var
minLength: Int\n open var name: String\n open var placeholder: String\n open var readOnly: Boolean\n open
var required: Boolean\n open var rows: Int\n open var wrap: String\n open val type: String\n open var
defaultValue: String\n open var value: String\n open val textLength: Int\n open val willValidate: Boolean\n
open val validity: ValidityState\n open val validationMessage: String\n open val labels: NodeList\n open var
selectionStart: Int?\n open var selectionEnd: Int?\n open var selectionDirection: String?\n fun checkValidity():
Boolean\n fun reportValidity(): Boolean\n fun setCustomValidity(error: String)\n fun select()\n fun
setRangeText(replacement: String)\n fun setRangeText(replacement: String, start: Int, end: Int, selectionMode:
SelectionMode = definedExternally)\n fun setSelectionRange(start: Int, end: Int, direction: String =
definedExternally)\n\n companion object {\n val ELEMENT_NODE: Short\n val ATTRIBUTE_NODE:
Short\n val TEXT_NODE: Short\n val CDATA_SECTION_NODE: Short\n val
ENTITY_REFERENCE_NODE: Short\n val ENTITY_NODE: Short\n val
PROCESSING_INSTRUCTION_NODE: Short\n val COMMENT_NODE: Short\n val
DOCUMENT_NODE: Short\n val DOCUMENT_TYPE_NODE: Short\n val
DOCUMENT_FRAGMENT_NODE: Short\n val NOTATION_NODE: Short\n val
DOCUMENT_POSITION_DISCONNECTED: Short\n val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n }\n}\n\n/**\n * Exposes the JavaScript
[HTMLKeygenElement](https://developer.mozilla.org/en/docs/Web/API/HTMLKeygenElement) to Kotlin\n
*\npublic external abstract class HTMLKeygenElement : HTMLInputElement {\n open var autofocus: Boolean\n
open var challenge: String\n open var disabled: Boolean\n open val form: HTMLFormElement?\n open var
keytype: String\n open var name: String\n open val type: String\n open val willValidate: Boolean\n open val
validity: ValidityState\n open val validationMessage: String\n open val labels: NodeList\n fun checkValidity():
Boolean\n fun reportValidity(): Boolean\n fun setCustomValidity(error: String)\n\n companion object {\n
val ELEMENT_NODE: Short\n val ATTRIBUTE_NODE: Short\n val TEXT_NODE: Short\n val
CDATA_SECTION_NODE: Short\n val ENTITY_REFERENCE_NODE: Short\n val ENTITY_NODE:
Short\n val PROCESSING_INSTRUCTION_NODE: Short\n val COMMENT_NODE: Short\n val
DOCUMENT_NODE: Short\n val DOCUMENT_TYPE_NODE: Short\n val
DOCUMENT_FRAGMENT_NODE: Short\n val NOTATION_NODE: Short\n val
DOCUMENT_POSITION_DISCONNECTED: Short\n val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n val DOCUMENT_POSITION_CONTAINS: Short\n

```

```

    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLInputElement](https://developer.mozilla.org/en/docs/Web/API/HTMLInputElement) to Kotlin\n
*/\npublic external abstract class HTMLInputElement : HTMLElement {\n    open val htmlFor: DOMTokenList\n
open val form: HTMLFormElement?\n    open var name: String\n    open val type: String\n    open var
defaultValue: String\n    open var value: String\n    open val willValidate: Boolean\n    open val validity:
ValidityState\n    open val validationMessage: String\n    open val labels: NodeList\n    fun checkValidity():
Boolean\n    fun reportValidity(): Boolean\n    fun setCustomValidity(error: String)\n\n    companion object {\n
val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val
CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE:
Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLProgressElement](https://developer.mozilla.org/en/docs/Web/API/HTMLProgressElement) to Kotlin\n
*/\npublic external abstract class HTMLProgressElement : HTMLElement {\n    open var value: Double\n    open
var max: Double\n    open val position: Double\n    open val labels: NodeList\n\n    companion object {\n
val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val
CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE:
Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLMeterElement](https://developer.mozilla.org/en/docs/Web/API/HTMLMeterElement) to Kotlin\n
*/\npublic external abstract class HTMLMeterElement : HTMLElement {\n    open var value: Double\n    open var min:
Double\n    open var max: Double\n    open var low: Double\n    open var high: Double\n    open var optimum:
Double\n    open val labels: NodeList\n\n    companion object {\n
val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val
CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n
val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLFieldSetElement](https://developer.mozilla.org/en/docs/Web/API/HTMLFieldSetElement) to Kotlin\n
*/\npublic external abstract class HTMLFieldSetElement : HTMLElement {\n    open var disabled: Boolean\n
open val form: HTMLFormElement?\n    open var name: String\n    open val type: String\n    open val elements:
HTMLCollection\n    open val willValidate: Boolean\n    open val validity: ValidityState\n    open val
validationMessage: String\n    fun checkValidity(): Boolean\n    fun reportValidity(): Boolean\n    fun
setCustomValidity(error: String)\n\n    companion object {\n
val ELEMENT_NODE: Short\n    val

```

```

ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }n}\n\n/**\n * Exposes the JavaScript
[HTMLLegendElement](https://developer.mozilla.org/en/docs/Web/API/HTMLLegendElement) to Kotlin\n
*/\npublic external abstract class HTMLLegendElement : HTMLInputElement {\n    open val form:
HTMLFormElement?\n    open var align: String\n\n    companion object {\n        val ELEMENT_NODE: Short\n
        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n
        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val
PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }n}\n\n/**\n * Exposes the JavaScript
[ValidityState](https://developer.mozilla.org/en/docs/Web/API/ValidityState) to Kotlin\n
*/\npublic external
abstract class ValidityState {\n    open val valueMissing: Boolean\n    open val typeMismatch: Boolean\n    open val
patternMismatch: Boolean\n    open val tooLong: Boolean\n    open val tooShort: Boolean\n    open val
rangeUnderflow: Boolean\n    open val rangeOverflow: Boolean\n    open val stepMismatch: Boolean\n    open val
badInput: Boolean\n    open val customError: Boolean\n    open val valid: Boolean\n}\n\n/**\n * Exposes the
JavaScript [HTMLDetailsElement](https://developer.mozilla.org/en/docs/Web/API/HTMLDetailsElement) to
Kotlin\n
*/\npublic external abstract class HTMLDetailsElement : HTMLInputElement {\n    open var open: Boolean\n
    companion object {\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n       
        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE:
Short\n        val ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val
COMMENT_NODE: Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n
        val DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }n}\n\npublic external abstract class
HTMLMenuElement : HTMLInputElement {\n    open var type: String\n    open var label: String\n    open var compact:
Boolean\n\n    companion object {\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n
        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE:
Short\n        val ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val
COMMENT_NODE: Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n
        val DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }n}\n\npublic external abstract class
HTMLMenuItemElement : HTMLInputElement {\n    open var type: String\n    open var label: String\n    open var icon:

```

```

String\n    open var disabled: Boolean\n    open var checked: Boolean\n    open var radiogroup: String\n    open var
default: Boolean\n\n    companion object {\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE:
Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val
ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val
PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n\n    public external open class
RelatedEvent(type: String, eventInitDict: RelatedEventInit = definedExternally) : Event {\n    open val
relatedTarget: EventTarget?\n\n    companion object {\n        val NONE: Short\n        val CAPTURING_PHASE:
Short\n        val AT_TARGET: Short\n        val BUBBLING_PHASE: Short\n    }\n\n    public external interface
RelatedEventInit : EventInit {\n        var relatedTarget: EventTarget? /* = null */\n        get() = definedExternally\n
set(value) = definedExternally\n    }\n\n    @Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n    @kotlin.internal.InlineOnly\n    public inline fun RelatedEventInit(relatedTarget:
EventTarget? = null, bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false):
RelatedEventInit {\n        val o = js("{}")\n        o["relatedTarget"] = relatedTarget\n        o["bubbles"] = bubbles\n
o["cancelable"] = cancelable\n        o["composed"] = composed\n        return o\n    }\n\n    /**\n     * Exposes the JavaScript
[HTMLDialogElement](https://developer.mozilla.org/en/docs/Web/API/HTMLDialogElement) to Kotlin\n
     */\n    public external abstract class HTMLDialogElement : HTMLInputElement {\n        open var open: Boolean\n        open var
returnValue: String\n        fun show(anchor: UnionElementOrMouseEvent = definedExternally)\n        fun
showModal(anchor: UnionElementOrMouseEvent = definedExternally)\n        fun close(returnValue: String =
definedExternally)\n\n        companion object {\n            val ELEMENT_NODE: Short\n            val ATTRIBUTE_NODE:
Short\n            val TEXT_NODE: Short\n            val CDATA_SECTION_NODE: Short\n            val
ENTITY_REFERENCE_NODE: Short\n            val ENTITY_NODE: Short\n            val
PROCESSING_INSTRUCTION_NODE: Short\n            val COMMENT_NODE: Short\n            val
DOCUMENT_NODE: Short\n            val DOCUMENT_TYPE_NODE: Short\n            val
DOCUMENT_FRAGMENT_NODE: Short\n            val NOTATION_NODE: Short\n            val
DOCUMENT_POSITION_DISCONNECTED: Short\n            val DOCUMENT_POSITION_PRECEDING: Short\n
            val DOCUMENT_POSITION_FOLLOWING: Short\n            val DOCUMENT_POSITION_CONTAINS: Short\n
            val DOCUMENT_POSITION_CONTAINED_BY: Short\n            val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n        }\n\n        /**\n         * Exposes the JavaScript
[HTMLScriptElement](https://developer.mozilla.org/en/docs/Web/API/HTMLScriptElement) to Kotlin\n
         */\n        public external abstract class HTMLScriptElement : HTMLInputElement, HTMLScriptElement {\n            open var src:
String\n            open var type: String\n            open var charset: String\n            open var async: Boolean\n            open var defer:
Boolean\n            open var crossOrigin: String?\n            open var text: String\n            open var nonce: String\n            open var event:
String\n            open var htmlFor: String\n\n            companion object {\n                val ELEMENT_NODE: Short\n                val
ATTRIBUTE_NODE: Short\n                val TEXT_NODE: Short\n                val CDATA_SECTION_NODE: Short\n                val
ENTITY_REFERENCE_NODE: Short\n                val ENTITY_NODE: Short\n                val
PROCESSING_INSTRUCTION_NODE: Short\n                val COMMENT_NODE: Short\n                val
DOCUMENT_NODE: Short\n                val DOCUMENT_TYPE_NODE: Short\n                val
DOCUMENT_FRAGMENT_NODE: Short\n                val NOTATION_NODE: Short\n                val
DOCUMENT_POSITION_DISCONNECTED: Short\n                val DOCUMENT_POSITION_PRECEDING: Short\n
                val DOCUMENT_POSITION_FOLLOWING: Short\n                val DOCUMENT_POSITION_CONTAINS: Short\n
                val DOCUMENT_POSITION_CONTAINED_BY: Short\n                val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n            }\n\n            /**\n             * Exposes the JavaScript

```

```

[HTMLTemplateElement](https://developer.mozilla.org/en/docs/Web/API/HTMLTemplateElement) to Kotlin
*  

\npublic external abstract class HTMLTemplateElement : HTMLElement {  

    open val content:  

    DocumentFragment companion object {  

        val ELEMENT_NODE: Short  

        val ATTRIBUTE_NODE:  

        Short  

        val TEXT_NODE: Short  

        val CDATA_SECTION_NODE: Short  

        val  

        ENTITY_REFERENCE_NODE: Short  

        val ENTITY_NODE: Short  

        val  

        PROCESSING_INSTRUCTION_NODE: Short  

        val COMMENT_NODE: Short  

        val  

        DOCUMENT_NODE: Short  

        val DOCUMENT_TYPE_NODE: Short  

        val  

        DOCUMENT_FRAGMENT_NODE: Short  

        val NOTATION_NODE: Short  

        val  

        DOCUMENT_POSITION_DISCONNECTED: Short  

        val DOCUMENT_POSITION_PRECEDING: Short  

        val  

        DOCUMENT_POSITION_FOLLOWING: Short  

        val DOCUMENT_POSITION_CONTAINS: Short  

        val  

        DOCUMENT_POSITION_CONTAINED_BY: Short  

        val  

        DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short  

    }  

}
\n\n/**  

 * Exposes the JavaScript  

[HTMLSlotElement](https://developer.mozilla.org/en/docs/Web/API/HTMLSlotElement) to Kotlin
*  

\npublic  

external abstract class HTMLSlotElement : HTMLElement {  

    open var name: String  

    fun  

    assignedNodes(options: AssignedNodesOptions = definedExternally): Array<Node>  

    companion object {  

        val ELEMENT_NODE: Short  

        val ATTRIBUTE_NODE: Short  

        val TEXT_NODE: Short  

        val  

        CDATA_SECTION_NODE: Short  

        val ENTITY_REFERENCE_NODE: Short  

        val ENTITY_NODE:  

        Short  

        val PROCESSING_INSTRUCTION_NODE: Short  

        val COMMENT_NODE: Short  

        val  

        DOCUMENT_NODE: Short  

        val DOCUMENT_TYPE_NODE: Short  

        val  

        DOCUMENT_FRAGMENT_NODE: Short  

        val NOTATION_NODE: Short  

        val  

        DOCUMENT_POSITION_DISCONNECTED: Short  

        val DOCUMENT_POSITION_PRECEDING: Short  

        val  

        DOCUMENT_POSITION_FOLLOWING: Short  

        val DOCUMENT_POSITION_CONTAINS: Short  

        val  

        DOCUMENT_POSITION_CONTAINED_BY: Short  

        val  

        DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short  

    }  

}
\n\npublic external interface  

AssignedNodesOptions {  

    var flatten: Boolean? /* = false */  

    get() = definedExternally  

    set(value) =  

    definedExternally  

}
\n\n@Suppress("INVISIBLE_REFERENCE",  

"INVISIBLE_MEMBER")  

\n@kotlin.internal.InlineOnly  

\npublic inline fun AssignedNodesOptions(flatten:  

Boolean? = false): AssignedNodesOptions {  

    val o = js("{}")  

    o["flatten"] = flatten  

    return  

o  

}
\n\n/**  

 * Exposes the JavaScript  

[HTMLCanvasElement](https://developer.mozilla.org/en/docs/Web/API/HTMLCanvasElement) to Kotlin
*  

\npublic external abstract class HTMLCanvasElement : HTMLImageElement, CanvasImageSource, TexImageSource  

{  

    open var width: Int  

    open var height: Int  

    fun getContext(contextId: String, vararg arguments: Any?):  

    RenderingContext?  

    fun toDataURL(type: String = definedExternally, quality: Any? = definedExternally):  

    String  

    fun toBlob(_callback: (Blob?) -> Unit, type: String = definedExternally, quality: Any? =  

    definedExternally)  

    companion object {  

        val ELEMENT_NODE: Short  

        val ATTRIBUTE_NODE:  

        Short  

        val TEXT_NODE: Short  

        val CDATA_SECTION_NODE: Short  

        val  

        ENTITY_REFERENCE_NODE: Short  

        val ENTITY_NODE: Short  

        val  

        PROCESSING_INSTRUCTION_NODE: Short  

        val COMMENT_NODE: Short  

        val  

        DOCUMENT_NODE: Short  

        val DOCUMENT_TYPE_NODE: Short  

        val  

        DOCUMENT_FRAGMENT_NODE: Short  

        val NOTATION_NODE: Short  

        val  

        DOCUMENT_POSITION_DISCONNECTED: Short  

        val DOCUMENT_POSITION_PRECEDING: Short  

        val  

        DOCUMENT_POSITION_FOLLOWING: Short  

        val DOCUMENT_POSITION_CONTAINS: Short  

        val  

        DOCUMENT_POSITION_CONTAINED_BY: Short  

        val  

        DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short  

    }  

}
\n\npublic external interface  

CanvasRenderingContext2DSettings {  

    var alpha: Boolean? /* = true */  

    get() = definedExternally  

    set(value) = definedExternally  

}
\n\n@Suppress("INVISIBLE_REFERENCE",  

"INVISIBLE_MEMBER")  

\n@kotlin.internal.InlineOnly  

\npublic inline fun  

CanvasRenderingContext2DSettings(alpha: Boolean? = true): CanvasRenderingContext2DSettings {  

    val o =

```

```

js("{}")\n o["alpha"] = alpha\n return o\n\n**\n * Exposes the JavaScript
[CanvasRenderingContext2D](https://developer.mozilla.org/en/docs/Web/API/CanvasRenderingContext2D) to
Kotlin\n */\npublic external abstract class CanvasRenderingContext2D : CanvasState, CanvasTransform,
CanvasCompositing, CanvasImageSmoothing, CanvasFillStrokeStyles, CanvasShadowStyles, CanvasFilters,
CanvasRect, CanvasDrawPath, CanvasUserInterface, CanvasText, CanvasDrawImage, CanvasHitRegion,
CanvasImageData, CanvasPathDrawingStyles, CanvasTextDrawingStyles, CanvasPath, RenderingContext {\n
open val canvas: HTMLCanvasElement\n}\n\npublic external interface CanvasState {\n fun save()\n fun
restore()\n}\n\npublic external interface CanvasTransform {\n fun scale(x: Double, y: Double)\n fun
rotate(angle: Double)\n fun translate(x: Double, y: Double)\n fun transform(a: Double, b: Double, c: Double, d:
Double, e: Double, f: Double)\n fun getTransform(): DOMMatrix\n fun setTransform(a: Double, b: Double, c:
Double, d: Double, e: Double, f: Double)\n fun setTransform(transform: dynamic = definedExternally)\n fun
resetTransform()\n}\n\npublic external interface CanvasCompositing {\n var globalAlpha: Double\n var
globalCompositeOperation: String\n}\n\npublic external interface CanvasImageSmoothing {\n var
imageSmoothingEnabled: Boolean\n var imageSmoothingQuality: ImageSmoothingQuality\n}\n\npublic external
interface CanvasFillStrokeStyles {\n var strokeStyle: dynamic\n get() = definedExternally\n set(value) =
definedExternally\n var fillStyle: dynamic\n get() = definedExternally\n set(value) = definedExternally\n
fun createLinearGradient(x0: Double, y0: Double, x1: Double, y1: Double): CanvasGradient\n fun
createRadialGradient(x0: Double, y0: Double, r0: Double, x1: Double, y1: Double, r1: Double): CanvasGradient\n
fun createPattern(image: CanvasImageSource, repetition: String): CanvasPattern?\n}\n\npublic external interface
CanvasShadowStyles {\n var shadowOffsetX: Double\n var shadowOffsetY: Double\n var shadowBlur:
Double\n var shadowColor: String\n}\n\npublic external interface CanvasFilters {\n var filter:
String\n}\n\npublic external interface CanvasRect {\n fun clearRect(x: Double, y: Double, w: Double, h:
Double)\n fun fillRect(x: Double, y: Double, w: Double, h: Double)\n fun strokeRect(x: Double, y: Double, w:
Double, h: Double)\n}\n\npublic external interface CanvasDrawPath {\n fun beginPath()\n fun fill(fillRule:
CanvasFillRule = definedExternally)\n fun fill(path: Path2D, fillRule: CanvasFillRule = definedExternally)\n
fun stroke()\n fun stroke(path: Path2D)\n fun clip(fillRule: CanvasFillRule = definedExternally)\n fun
clip(path: Path2D, fillRule: CanvasFillRule = definedExternally)\n fun resetClip()\n fun isPointInPath(x:
Double, y: Double, fillRule: CanvasFillRule = definedExternally): Boolean\n fun isPointInPath(path: Path2D, x:
Double, y: Double, fillRule: CanvasFillRule = definedExternally): Boolean\n fun isPointInStroke(x: Double, y:
Double): Boolean\n fun isPointInStroke(path: Path2D, x: Double, y: Double): Boolean\n}\n\npublic external
interface CanvasUserInterface {\n fun drawFocusIfNeeded(element: Element)\n fun drawFocusIfNeeded(path:
Path2D, element: Element)\n fun scrollPathIntoView()\n fun scrollPathIntoView(path: Path2D)\n}\n\npublic
external interface CanvasText {\n fun fillText(text: String, x: Double, y: Double, maxWidth: Double =
definedExternally)\n fun strokeText(text: String, x: Double, y: Double, maxWidth: Double = definedExternally)\n
fun measureText(text: String): TextMetrics\n}\n\npublic external interface CanvasDrawImage {\n fun
drawImage(image: CanvasImageSource, dx: Double, dy: Double)\n fun drawImage(image: CanvasImageSource,
dx: Double, dy: Double, dw: Double, dh: Double)\n fun drawImage(image: CanvasImageSource, sx: Double, sy:
Double, sw: Double, sh: Double, dx: Double, dy: Double, dw: Double, dh: Double)\n}\n\npublic external interface
CanvasHitRegion {\n fun addHitRegion(options: HitRegionOptions = definedExternally)\n fun
removeHitRegion(id: String)\n fun clearHitRegions()\n}\n\npublic external interface CanvasImageData {\n fun
createImageData(sw: Double, sh: Double): ImageData\n fun createImageData(imagedata: ImageData):
ImageData\n fun getImageData(sx: Double, sy: Double, sw: Double, sh: Double): ImageData\n fun
putImageData(imagedata: ImageData, dx: Double, dy: Double)\n fun putImageData(imagedata: ImageData, dx:
Double, dy: Double, dirtyX: Double, dirtyY: Double, dirtyWidth: Double, dirtyHeight: Double)\n}\n\npublic
external interface CanvasPathDrawingStyles {\n var lineWidth: Double\n var lineCap: CanvasLineCap\n var
lineJoin: CanvasLineJoin\n var miterLimit: Double\n var lineDashOffset: Double\n fun setLineDash(segments:
Array<Double>)\n fun getLineDash(): Array<Double>\n}\n\npublic external interface CanvasTextDrawingStyles
{\n var font: String\n var textAlign: CanvasTextAlign\n var textBaseline: CanvasTextBaseline\n var

```

```

direction: CanvasDirection\n}\n\npublic external interface CanvasPath {\n  fun closePath()\n  fun moveTo(x:
Double, y: Double)\n  fun lineTo(x: Double, y: Double)\n  fun quadraticCurveTo(cpx: Double, cpy: Double, x:
Double, y: Double)\n  fun bezierCurveTo(cp1x: Double, cp1y: Double, cp2x: Double, cp2y: Double, x: Double, y:
Double)\n  fun arcTo(x1: Double, y1: Double, x2: Double, y2: Double, radius: Double)\n  fun arcTo(x1: Double,
y1: Double, x2: Double, y2: Double, radiusX: Double, radiusY: Double, rotation: Double)\n  fun rect(x: Double, y:
Double, w: Double, h: Double)\n  fun arc(x: Double, y: Double, radius: Double, startAngle: Double, endAngle:
Double, anticlockwise: Boolean = definedExternally)\n  fun ellipse(x: Double, y: Double, radiusX: Double,
radiusY: Double, rotation: Double, startAngle: Double, endAngle: Double, anticlockwise: Boolean =
definedExternally)\n}\n\n/**\n * Exposes the JavaScript
[CanvasGradient](https://developer.mozilla.org/en/docs/Web/API/CanvasGradient) to Kotlin\n */\n\npublic external
abstract class CanvasGradient {\n  fun addColorStop(offset: Double, color: String)\n}\n\n/**\n * Exposes the
JavaScript [CanvasPattern](https://developer.mozilla.org/en/docs/Web/API/CanvasPattern) to Kotlin\n */\n\npublic
external abstract class CanvasPattern {\n  fun setTransform(transform: dynamic = definedExternally)\n}\n\n/**\n *
Exposes the JavaScript [TextMetrics](https://developer.mozilla.org/en/docs/Web/API/TextMetrics) to Kotlin\n
*/\n\npublic external abstract class TextMetrics {\n  open val width: Double\n  open val actualBoundingBoxLeft:
Double\n  open val actualBoundingBoxRight: Double\n  open val fontBoundingBoxAscent: Double\n  open val
fontBoundingBoxDescent: Double\n  open val actualBoundingBoxAscent: Double\n  open val
actualBoundingBoxDescent: Double\n  open val emHeightAscent: Double\n  open val emHeightDescent:
Double\n  open val hangingBaseline: Double\n  open val alphabeticBaseline: Double\n  open val
ideographicBaseline: Double\n}\n\npublic external interface HitRegionOptions {\n  var path: Path2D? /* = null
*/\n  get() = definedExternally\n  set(value) = definedExternally\n  var fillRule: CanvasFillRule? /* =
CanvasFillRule.NONZERO */\n  get() = definedExternally\n  set(value) = definedExternally\n  var id:
String? /* = \"\" */\n  get() = definedExternally\n  set(value) = definedExternally\n  var parentID: String? /*
= null */\n  get() = definedExternally\n  set(value) = definedExternally\n  var cursor: String? /* = \"inherit\"
*/\n  get() = definedExternally\n  set(value) = definedExternally\n  var control: Element? /* = null */\n
get() = definedExternally\n  set(value) = definedExternally\n  var label: String? /* = null */\n  get() =
definedExternally\n  set(value) = definedExternally\n  var role: String? /* = null */\n  get() =
definedExternally\n  set(value) = definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun HitRegionOptions(path: Path2D? =
null, fillRule: CanvasFillRule? = CanvasFillRule.NONZERO, id: String? = \"\", parentID: String? = null, cursor:
String? = \"inherit\", control: Element? = null, label: String? = null, role: String? = null): HitRegionOptions {\n
  val o = js(\"{\}\")\n  o[\"path\"] = path\n  o[\"fillRule\"] = fillRule\n  o[\"id\"] = id\n  o[\"parentID\"] = parentID\n
  o[\"cursor\"] = cursor\n  o[\"control\"] = control\n  o[\"label\"] = label\n  o[\"role\"] = role\n  return
o\n}\n\n/**\n * Exposes the JavaScript [ImageData](https://developer.mozilla.org/en/docs/Web/API/ImageData) to
Kotlin\n */\n\npublic external open class ImageData : ImageBitmapSource, TexImageSource {\n  constructor(sw:
Int, sh: Int)\n  constructor(data: Uint8ClampedArray, sw: Int, sh: Int = definedExternally)\n  open val width: Int\n
open val height: Int\n  open val data: Uint8ClampedArray\n}\n\n/**\n * Exposes the JavaScript
[Path2D](https://developer.mozilla.org/en/docs/Web/API/Path2D) to Kotlin\n */\n\npublic external open class
Path2D() : CanvasPath {\n  constructor(path: Path2D)\n  constructor(paths: Array<Path2D>, fillRule:
CanvasFillRule = definedExternally)\n  constructor(d: String)\n  fun addPath(path: Path2D, transform: dynamic =
definedExternally)\n  override fun closePath()\n  override fun moveTo(x: Double, y: Double)\n  override fun
lineTo(x: Double, y: Double)\n  override fun quadraticCurveTo(cpx: Double, cpy: Double, x: Double, y: Double)\n
  override fun bezierCurveTo(cp1x: Double, cp1y: Double, cp2x: Double, cp2y: Double, x: Double, y: Double)\n
  override fun arcTo(x1: Double, y1: Double, x2: Double, y2: Double, radius: Double)\n  override fun arcTo(x1:
Double, y1: Double, x2: Double, y2: Double, radiusX: Double, radiusY: Double, rotation: Double)\n  override fun
rect(x: Double, y: Double, w: Double, h: Double)\n  override fun arc(x: Double, y: Double, radius: Double,
startAngle: Double, endAngle: Double, anticlockwise: Boolean /* = definedExternally */) \n  override fun ellipse(x:
Double, y: Double, radiusX: Double, radiusY: Double, rotation: Double, startAngle: Double, endAngle: Double,

```

```

anticlockwise: Boolean /* = definedExternally */)\n}\n\n/**\n * Exposes the JavaScript
[ImageBitmapRenderingContext](https://developer.mozilla.org/en/docs/Web/API/ImageBitmapRenderingContext)
to Kotlin\n *\npublic external abstract class ImageBitmapRenderingContext {\n  open val canvas:
HTMLCanvasElement\n  fun transferFromImageBitmap(bitmap: ImageBitmap?)\n}\n\npublic external interface
ImageBitmapRenderingContextSettings {\n  var alpha: Boolean? /* = true */\n  get() = definedExternally\n  set(value) = definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun
ImageBitmapRenderingContextSettings(alpha: Boolean? = true): ImageBitmapRenderingContextSettings {\n  val o
= js("{}")\n  o["alpha"] = alpha\n  return o\n}\n\n/**\n * Exposes the JavaScript
[CustomElementRegistry](https://developer.mozilla.org/en/docs/Web/API/CustomElementRegistry) to Kotlin\n
*\npublic external abstract class CustomElementRegistry {\n  fun define(name: String, constructor: () -> dynamic,
options: ElementDefinitionOptions = definedExternally)\n  fun get(name: String): Any?\n  fun
whenDefined(name: String): Promise<Unit>\n}\n\npublic external interface ElementDefinitionOptions {\n  var
extends: String?\n  get() = definedExternally\n  set(value) =
definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun ElementDefinitionOptions(extends:
String? = undefined): ElementDefinitionOptions {\n  val o = js("{}")\n  o["extends"] = extends\n  return
o\n}\n\npublic external interface ElementContentEditable {\n  var contentEditable: String\n  val
isContentEditable: Boolean\n}\n\n/**\n * Exposes the JavaScript
[DataTransfer](https://developer.mozilla.org/en/docs/Web/API/DataTransfer) to Kotlin\n *\npublic external
abstract class DataTransfer {\n  open var dropEffect: String\n  open var effectAllowed: String\n  open val items:
DataTransferItemList\n  open val types: Array<out String>\n  open val files: FileList\n  fun
setDragImage(image: Element, x: Int, y: Int)\n  fun getData(format: String): String\n  fun setData(format: String,
data: String)\n  fun clearData(format: String = definedExternally)\n}\n\n/**\n * Exposes the JavaScript
[DataTransferItemList](https://developer.mozilla.org/en/docs/Web/API/DataTransferItemList) to Kotlin\n *\npublic
external abstract class DataTransferItemList {\n  open val length: Int\n  fun add(data: String, type: String):
DataTransferItem?\n  fun add(data: File): DataTransferItem?\n  fun remove(index: Int)\n  fun
clear()\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun
DataTransferItemList.get(index: Int): DataTransferItem? = asDynamic()[index]\n\n/**\n * Exposes the JavaScript
[DataTransferItem](https://developer.mozilla.org/en/docs/Web/API/DataTransferItem) to Kotlin\n *\npublic
external abstract class DataTransferItem {\n  open val kind: String\n  open val type: String\n  fun
getAsString(_callback: ((String) -> Unit)?)\n  fun getAsFile(): File?\n}\n\n/**\n * Exposes the JavaScript
[DragEvent](https://developer.mozilla.org/en/docs/Web/API/TouchEvent) to Kotlin\n *\npublic external open class
DragEvent(type: String, eventInitDict: DragEventInit = definedExternally) : MouseEvent {\n  open val
dataTransfer: DataTransfer?\n\n  companion object {\n    val NONE: Short\n    val CAPTURING_PHASE:
Short\n    val AT_TARGET: Short\n    val BUBBLING_PHASE: Short\n  }\n}\n\npublic external interface
DragEventInit : MouseEventInit {\n  var dataTransfer: DataTransfer? /* = null */\n  get() = definedExternally\n
  set(value) = definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun DragEventInit(dataTransfer:
DataTransfer? = null, screenX: Int? = 0, screenY: Int? = 0, clientX: Int? = 0, clientY: Int? = 0, button: Short? = 0,
buttons: Short? = 0, relatedTarget: EventTarget? = null, region: String? = null, ctrlKey: Boolean? = false, shiftKey:
Boolean? = false, altKey: Boolean? = false, metaKey: Boolean? = false, modifierAltGraph: Boolean? = false,
modifierCapsLock: Boolean? = false, modifierFn: Boolean? = false, modifierFnLock: Boolean? = false,
modifierHyper: Boolean? = false, modifierNumLock: Boolean? = false, modifierScrollLock: Boolean? = false,
modifierSuper: Boolean? = false, modifierSymbol: Boolean? = false, modifierSymbolLock: Boolean? = false, view:
Window? = null, detail: Int? = 0, bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? =
false): DragEventInit {\n  val o = js("{}")\n  o["dataTransfer"] = dataTransfer\n  o["screenX"] = screenX\n

```

```

o["screenY"] = screenY\n o["clientX"] = clientX\n o["clientY"] = clientY\n o["button"] = button\n
o["buttons"] = buttons\n o["relatedTarget"] = relatedTarget\n o["region"] = region\n o["ctrlKey"] =
ctrlKey\n o["shiftKey"] = shiftKey\n o["altKey"] = altKey\n o["metaKey"] = metaKey\n
o["modifierAltGraph"] = modifierAltGraph\n o["modifierCapsLock"] = modifierCapsLock\n
o["modifierFn"] = modifierFn\n o["modifierFnLock"] = modifierFnLock\n o["modifierHyper"] =
modifierHyper\n o["modifierNumLock"] = modifierNumLock\n o["modifierScrollLock"] =
modifierScrollLock\n o["modifierSuper"] = modifierSuper\n o["modifierSymbol"] = modifierSymbol\n
o["modifierSymbolLock"] = modifierSymbolLock\n o["view"] = view\n o["detail"] = detail\n
o["bubbles"] = bubbles\n o["cancelable"] = cancelable\n o["composed"] = composed\n return
o\n}\n\n/**\n * Exposes the JavaScript [Window](https://developer.mozilla.org/en/docs/Web/API/Window) to
Kotlin\n *\npublic external abstract class Window : EventTarget, GlobalEventHandlers, WindowEventHandlers,
WindowOrWorkerGlobalScope, WindowSessionStorage, WindowLocalStorage, GlobalPerformance,
UnionMessagePortOrWindowProxy {\n open val window: Window\n open val self: Window\n open val
document: Document\n open var name: String\n open val location: Location\n open val history: History\n
open val customElements: CustomElementRegistry\n open val locationbar: BarProp\n open val menubar:
BarProp\n open val personalbar: BarProp\n open val scrollbars: BarProp\n open val statusbar: BarProp\n
open val toolbar: BarProp\n open var status: String\n open val closed: Boolean\n open val frames: Window\n
open val length: Int\n open val top: Window\n open var opener: Any?\n open val parent: Window\n open val
frameElement: Element?\n open val navigator: Navigator\n open val applicationCache: ApplicationCache\n
open val external: External\n open val screen: Screen\n open val innerWidth: Int\n open val innerHeight: Int\n
open val scrollX: Double\n open val pageXOffset: Double\n open val scrollY: Double\n open val
pageYOffset: Double\n open val screenX: Int\n open val screenY: Int\n open val outerWidth: Int\n open val
outerHeight: Int\n open val devicePixelRatio: Double\n fun close()\n fun stop()\n fun focus()\n fun blur()\n
fun open(url: String = definedExternally, target: String = definedExternally, features: String = definedExternally):
Window?\n fun alert()\n fun alert(message: String)\n fun confirm(message: String = definedExternally):
Boolean\n fun prompt(message: String = definedExternally, default: String = definedExternally): String?\n fun
print()\n fun requestAnimationFrame(callback: (Double) -> Unit): Int\n fun cancelAnimationFrame(handle:
Int)\n fun postMessage(message: Any?, targetOrigin: String, transfer: Array<dynamic> = definedExternally)\n
fun captureEvents()\n fun releaseEvents()\n fun matchMedia(query: String): MediaQueryList\n fun moveTo(x:
Int, y: Int)\n fun moveBy(x: Int, y: Int)\n fun resizeTo(x: Int, y: Int)\n fun resizeBy(x: Int, y: Int)\n fun
scroll(options: ScrollToOptions = definedExternally)\n fun scroll(x: Double, y: Double)\n fun scrollTo(options:
ScrollToOptions = definedExternally)\n fun scrollTo(x: Double, y: Double)\n fun scrollBy(options:
ScrollToOptions = definedExternally)\n fun scrollBy(x: Double, y: Double)\n fun getComputedStyle(elt:
Element, pseudoElt: String? = definedExternally):
CSSStyleDeclaration\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun Window.get(name: String):
dynamic = asDynamic()[name]\n\npublic external abstract class BarProp {\n open val visible: Boolean\n}\n\n/**\n
* Exposes the JavaScript [History](https://developer.mozilla.org/en/docs/Web/API/History) to Kotlin\n *\npublic
external abstract class History {\n open val length: Int\n open var scrollRestoration: ScrollRestoration\n open
val state: Any?\n fun go(delta: Int = definedExternally)\n fun back()\n fun forward()\n fun pushState(data:
Any?, title: String, url: String? = definedExternally)\n fun replaceState(data: Any?, title: String, url: String? =
definedExternally)\n}\n\n/**\n * Exposes the JavaScript
[Location](https://developer.mozilla.org/en/docs/Web/API/Location) to Kotlin\n *\npublic external abstract class
Location {\n open var href: String\n open val origin: String\n open var protocol: String\n open var host:
String\n open var hostname: String\n open var port: String\n open var pathname: String\n open var search:
String\n open var hash: String\n open val ancestorOrigins: Array<out String>\n fun assign(url: String)\n fun
replace(url: String)\n fun reload()\n}\n\n/**\n * Exposes the JavaScript
[PopStateEvent](https://developer.mozilla.org/en/docs/Web/API/PopStateEvent) to Kotlin\n *\npublic external

```

```

open class PopStateEvent(type: String, eventInitDict: PopStateEventInit = definedExternally) : Event {
    open val state: Any?
    companion object {
        val NONE: Short
        val CAPTURING_PHASE: Short
        val AT_TARGET: Short
        val BUBBLING_PHASE: Short
    }
    public external interface PopStateEventInit : EventInit {
        var state: Any? /* = null */
        get() = definedExternally
        set(value) = definedExternally
    }
    @Suppress("INVISIBLE_REFERENCE", "INVISIBLE_MEMBER")
    @kotlin.internal.InlineOnly
    public inline fun PopStateEventInit(state: Any? = null, bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false): PopStateEventInit {
        val o = js("{}")
        o["state"] = state
        o["bubbles"] = bubbles
        o["cancelable"] = cancelable
        o["composed"] = composed
        return o
    }
    /** Exposes the JavaScript [HashChangeEvent](https://developer.mozilla.org/en/docs/Web/API/HashChangeEvent) to Kotlin */
    public external open class HashChangeEvent(type: String, eventInitDict: HashChangeEventInit = definedExternally) : Event {
        open val oldURL: String
        open val newURL: String
        companion object {
            val NONE: Short
            val CAPTURING_PHASE: Short
            val AT_TARGET: Short
            val BUBBLING_PHASE: Short
        }
        public external interface HashChangeEventInit : EventInit {
            var oldURL: String? /* = "" */
            get() = definedExternally
            set(value) = definedExternally
            var newURL: String? /* = "" */
            get() = definedExternally
            set(value) = definedExternally
        }
        @Suppress("INVISIBLE_REFERENCE", "INVISIBLE_MEMBER")
        @kotlin.internal.InlineOnly
        public inline fun HashChangeEventInit(oldURL: String? = "", newURL: String? = "", bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false): HashChangeEventInit {
            val o = js("{}")
            o["oldURL"] = oldURL
            o["newURL"] = newURL
            o["bubbles"] = bubbles
            o["cancelable"] = cancelable
            o["composed"] = composed
            return o
        }
        /** Exposes the JavaScript [PageTransitionEvent](https://developer.mozilla.org/en/docs/Web/API/PageTransitionEvent) to Kotlin */
        public external open class PageTransitionEvent(type: String, eventInitDict: PageTransitionEventInit = definedExternally) : Event {
            open val persisted: Boolean
            companion object {
                val NONE: Short
                val CAPTURING_PHASE: Short
                val AT_TARGET: Short
                val BUBBLING_PHASE: Short
            }
            public external interface PageTransitionEventInit : EventInit {
                var persisted: Boolean? /* = false */
                get() = definedExternally
                set(value) = definedExternally
            }
            @Suppress("INVISIBLE_REFERENCE", "INVISIBLE_MEMBER")
            @kotlin.internal.InlineOnly
            public inline fun PageTransitionEventInit(persisted: Boolean? = false, bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false): PageTransitionEventInit {
                val o = js("{}")
                o["persisted"] = persisted
                o["bubbles"] = bubbles
                o["cancelable"] = cancelable
                o["composed"] = composed
                return o
            }
            /** Exposes the JavaScript [BeforeUnloadEvent](https://developer.mozilla.org/en/docs/Web/API/BeforeUnloadEvent) to Kotlin */
            public external open class BeforeUnloadEvent : Event {
                var returnValue: String
                companion object {
                    val NONE: Short
                    val CAPTURING_PHASE: Short
                    val AT_TARGET: Short
                    val BUBBLING_PHASE: Short
                }
                public external abstract class ApplicationCache : EventTarget {
                    open val status: Short
                    open var onchecking: ((Event) -> dynamic)?
                    open var onerror: ((Event) -> dynamic)?
                    open var onnoupdate: ((Event) -> dynamic)?
                    open var ondownloading: ((Event) -> dynamic)?
                    open var onprogress: ((ProgressEvent) -> dynamic)?
                    open var onupdateready: ((Event) -> dynamic)?
                    open var oncached: ((Event) -> dynamic)?
                    open var onobsolete: ((Event) -> dynamic)?
                    fun update()
                    fun abort()
                    fun swapCache()
                }
                companion object {
                    val UNCACHED: Short
                    val IDLE: Short
                    val CHECKING: Short
                    val DOWNLOADING: Short
                    val UPDATEREADY: Short
                    val OBSOLETE: Short
                }
            }
            /** Exposes the JavaScript [NavigatorOnLine](https://developer.mozilla.org/en/docs/Web/API/NavigatorOnLine) to Kotlin */
            public external interface NavigatorOnLine {
                val onLine: Boolean
            }
            /** Exposes the JavaScript [ErrorEvent](https://developer.mozilla.org/en/docs/Web/API/ErrorEvent) to Kotlin */
            public external open class ErrorEvent(type: String, eventInitDict: ErrorEventInit = definedExternally) : Event {
                open val message: String
                open val filename: String
                open val lineno: Int
                open val colno: Int
                open val error: Any?
                companion object {
                    val NONE: Short
                    val CAPTURING_PHASE: Short
                    val AT_TARGET: Short
                    val BUBBLING_PHASE: Short
                }
            }

```

```

BUBBLING_PHASE: Short\n } \n\n\npublic external interface ErrorEventInit : EventInit {\n var message:
String? /* = \"\" */\n get() = definedExternally\n set(value) = definedExternally\n var filename: String? /*
= \"\" */\n get() = definedExternally\n set(value) = definedExternally\n var lineno: Int? /* = 0 */\n
get() = definedExternally\n set(value) = definedExternally\n var colno: Int? /* = 0 */\n get() =
definedExternally\n set(value) = definedExternally\n var error: Any? /* = null */\n get() =
definedExternally\n set(value) = definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun ErrorEventInit(message: String? = \"\",
filename: String? = \"\", lineno: Int? = 0, colno: Int? = 0, error: Any? = null, bubbles: Boolean? = false, cancelable:
Boolean? = false, composed: Boolean? = false): ErrorEventInit {\n val o = js(\"({})\")\n o[\"message\"] =
message\n o[\"filename\"] = filename\n o[\"lineno\"] = lineno\n o[\"colno\"] = colno\n o[\"error\"] = error\n
o[\"bubbles\"] = bubbles\n o[\"cancelable\"] = cancelable\n o[\"composed\"] = composed\n return
o\n}\n\n/**\n * Exposes the JavaScript
[PromiseRejectionEvent](https://developer.mozilla.org/en/docs/Web/API/PromiseRejectionEvent) to Kotlin\n
*/\npublic external open class PromiseRejectionEvent(type: String, eventInitDict: PromiseRejectionEventInit) :
Event {\n open val promise: Promise<Any?>\n open val reason: Any?\n\n companion object {\n val
NONE: Short\n val CAPTURING_PHASE: Short\n val AT_TARGET: Short\n val
BUBBLING_PHASE: Short\n } \n\n\npublic external interface PromiseRejectionEventInit : EventInit {\n var
promise: Promise<Any?>?\n var reason: Any?\n get() = definedExternally\n set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun PromiseRejectionEventInit(promise:
Promise<Any?>?, reason: Any? = undefined, bubbles: Boolean? = false, cancelable: Boolean? = false, composed:
Boolean? = false): PromiseRejectionEventInit {\n val o = js(\"({})\")\n o[\"promise\"] = promise\n
o[\"reason\"] = reason\n o[\"bubbles\"] = bubbles\n o[\"cancelable\"] = cancelable\n o[\"composed\"] =
composed\n return o\n}\n\n/**\n * Exposes the JavaScript
[GlobalEventHandlers](https://developer.mozilla.org/en/docs/Web/API/GlobalEventHandlers) to Kotlin\n
*/\npublic external interface GlobalEventHandlers {\n var onabort: ((Event) -> dynamic)?\n get() =
definedExternally\n set(value) = definedExternally\n var onblur: ((FocusEvent) -> dynamic)?\n get() =
definedExternally\n set(value) = definedExternally\n var oncancel: ((Event) -> dynamic)?\n get() =
definedExternally\n set(value) = definedExternally\n var oncanplay: ((Event) -> dynamic)?\n get() =
definedExternally\n set(value) = definedExternally\n var oncanplaythrough: ((Event) -> dynamic)?\n get()
= definedExternally\n set(value) = definedExternally\n var onchange: ((Event) -> dynamic)?\n get() =
definedExternally\n set(value) = definedExternally\n var onclick: ((MouseEvent) -> dynamic)?\n get() =
definedExternally\n set(value) = definedExternally\n var onclose: ((Event) -> dynamic)?\n get() =
definedExternally\n set(value) = definedExternally\n var oncontextmenu: ((MouseEvent) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var oncuechange: ((Event) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var ondblclick: ((MouseEvent) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var ondrag: ((DragEvent) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var ondragend: ((DragEvent) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var ondragenter: ((DragEvent) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var ondragexit: ((DragEvent) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var ondragleave: ((DragEvent) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var ondragover: ((DragEvent) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var ondragstart: ((DragEvent) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var ondrop: ((DragEvent) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var ondurationchange: ((Event) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var onemptied: ((Event) -> dynamic)?\n
get() = definedExternally\n set(value) = definedExternally\n var onended: ((Event) -> dynamic)?\n get() =
definedExternally\n set(value) = definedExternally\n var onerror: ((dynamic, String, Int, Int, Any?) ->

```



```

set(value) = definedExternally\n    var onpointerout: ((PointerEvent) -> dynamic)?\n    get() = definedExternally\n
    set(value) = definedExternally\n    var onpointerenter: ((PointerEvent) -> dynamic)?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var onpointerleave: ((PointerEvent) -> dynamic)?\n
get() = definedExternally\n    set(value) = definedExternally\n}\n\n**\n * Exposes the JavaScript
[WindowEventHandlers](https://developer.mozilla.org/en/docs/Web/API/WindowEventHandlers) to Kotlin\n
*\npublic external interface WindowEventHandlers {\n    var onafterprint: ((Event) -> dynamic)?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var onbeforeprint: ((Event) -> dynamic)?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var onbeforeunload: ((BeforeUnloadEvent) ->
String?)?\n    get() = definedExternally\n    set(value) = definedExternally\n    var onhashchange:
((HashChangeEvent) -> dynamic)?\n    get() = definedExternally\n    set(value) = definedExternally\n    var
onlanguagechange: ((Event) -> dynamic)?\n    get() = definedExternally\n    set(value) = definedExternally\n
var onmessage: ((MessageEvent) -> dynamic)?\n    get() = definedExternally\n    set(value) =
definedExternally\n    var onoffline: ((Event) -> dynamic)?\n    get() = definedExternally\n    set(value) =
definedExternally\n    var ononline: ((Event) -> dynamic)?\n    get() = definedExternally\n    set(value) =
definedExternally\n    var onpagehide: ((PageTransitionEvent) -> dynamic)?\n    get() = definedExternally\n
set(value) = definedExternally\n    var onpageshow: ((PageTransitionEvent) -> dynamic)?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var onpopstate: ((PopStateEvent) -> dynamic)?\n
get() = definedExternally\n    set(value) = definedExternally\n    var onrejectionhandled: ((Event) -> dynamic)?\n
    get() = definedExternally\n    set(value) = definedExternally\n    var onstorage: ((StorageEvent) -> dynamic)?\n
    get() = definedExternally\n    set(value) = definedExternally\n    var onunhandledrejection:
((PromiseRejectionEvent) -> dynamic)?\n    get() = definedExternally\n    set(value) = definedExternally\n
var onunload: ((Event) -> dynamic)?\n    get() = definedExternally\n    set(value) =
definedExternally\n}\n\npublic external interface DocumentAndElementEventHandlers {\n    var oncopy:
((ClipboardEvent) -> dynamic)?\n    get() = definedExternally\n    set(value) = definedExternally\n    var oncut:
((ClipboardEvent) -> dynamic)?\n    get() = definedExternally\n    set(value) = definedExternally\n    var
onpaste: ((ClipboardEvent) -> dynamic)?\n    get() = definedExternally\n    set(value) =
definedExternally\n}\n\n**\n * Exposes the JavaScript
[WindowOrWorkerGlobalScope](https://developer.mozilla.org/en/docs/Web/API/WindowOrWorkerGlobalScope)
to Kotlin\n
*\npublic external interface WindowOrWorkerGlobalScope {\n    val origin: String\n    val caches:
CacheStorage\n    fun btoa(data: String): String\n    fun atob(data: String): String\n    fun setTimeout(handler:
dynamic, timeout: Int = definedExternally, vararg arguments: Any?): Int\n    fun clearTimeout(handle: Int =
definedExternally)\n    fun setInterval(handler: dynamic, timeout: Int = definedExternally, vararg arguments: Any?):
Int\n    fun clearInterval(handle: Int = definedExternally)\n    fun createImageBitmap(image: ImageBitmapSource,
options: ImageBitmapOptions = definedExternally): Promise<ImageBitmap>\n    fun createImageBitmap(image:
ImageBitmapSource, sx: Int, sy: Int, sw: Int, sh: Int, options: ImageBitmapOptions = definedExternally):
Promise<ImageBitmap>\n    fun fetch(input: dynamic, init: RequestInit = definedExternally):
Promise<Response>\n}\n\n**\n * Exposes the JavaScript
[Navigator](https://developer.mozilla.org/en/docs/Web/API/Navigator) to Kotlin\n
*\npublic external abstract class
Navigator : NavigatorID, NavigatorLanguage, NavigatorOnLine, NavigatorContentUtils, NavigatorCookies,
NavigatorPlugins, NavigatorConcurrentHardware {\n    open val clipboard: Clipboard\n    open val mediaDevices:
MediaDevices\n    open val maxTouchPoints: Int\n    open val serviceWorker: ServiceWorkerContainer\n    fun
requestMediaKeySystemAccess(keySystem: String, supportedConfigurations:
Array<MediaKeySystemConfiguration>): Promise<MediaKeySystemAccess>\n    fun getUserMedia(constraints:
MediaStreamConstraints, successCallback: (MediaStream) -> Unit, errorCallback: (dynamic) -> Unit)\n    fun
vibrate(pattern: dynamic): Boolean\n}\n\n**\n * Exposes the JavaScript
[NavigatorID](https://developer.mozilla.org/en/docs/Web/API/NavigatorID) to Kotlin\n
*\npublic external interface
NavigatorID {\n    val appCodeName: String\n    val appName: String\n    val appVersion: String\n    val platform:
String\n    val product: String\n    val productSub: String\n    val userAgent: String\n    val vendor: String\n    val

```

```

vendorSub: String\n val oscpu: String\n fun taintEnabled(): Boolean\n}\n\n/**\n * Exposes the JavaScript
[NavigatorLanguage](https://developer.mozilla.org/en/docs/Web/API/NavigatorLanguage) to Kotlin\n */\npublic
external interface NavigatorLanguage {\n val language: String\n val languages: Array<out String>\n}\n\npublic
external interface NavigatorContentUtils {\n fun registerProtocolHandler(scheme: String, url: String, title:
String)\n fun registerContentHandler(mimeType: String, url: String, title: String)\n fun
isProtocolHandlerRegistered(scheme: String, url: String): String\n fun isContentHandlerRegistered(mimeType:
String, url: String): String\n fun unregisterProtocolHandler(scheme: String, url: String)\n fun
unregisterContentHandler(mimeType: String, url: String)\n}\n\npublic external interface NavigatorCookies {\n val
cookieEnabled: Boolean\n}\n\n/**\n * Exposes the JavaScript
[NavigatorPlugins](https://developer.mozilla.org/en/docs/Web/API/NavigatorPlugins) to Kotlin\n */\npublic
external interface NavigatorPlugins {\n val plugins: PluginArray\n val mimeTypes: MimeTypeError\n fun
javaEnabled(): Boolean\n}\n\n/**\n * Exposes the JavaScript
[PluginArray](https://developer.mozilla.org/en/docs/Web/API/PluginArray) to Kotlin\n */\npublic external abstract
class PluginArray : ItemArrayLike<Plugin> {\n fun refresh(reload: Boolean = definedExternally)\n override fun
item(index: Int): Plugin?\n fun namedItem(name: String):
Plugin?\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun PluginArray.get(index: Int):
Plugin? = asDynamic()[index]\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun PluginArray.get(name:
String): Plugin? = asDynamic()[name]\n\n/**\n * Exposes the JavaScript
[MimeTypeArray](https://developer.mozilla.org/en/docs/Web/API/MimeTypeArray) to Kotlin\n */\npublic external
abstract class MimeTypeArray : ItemArrayLike<MimeType> {\n override fun item(index: Int): MimeType?\n
fun namedItem(name: String): MimeType?\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun MimeTypeArray.get(index:
Int): MimeType? = asDynamic()[index]\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun MimeTypeArray.get(name:
String): MimeType? = asDynamic()[name]\n\n/**\n * Exposes the JavaScript
[Plugin](https://developer.mozilla.org/en/docs/Web/API/Plugin) to Kotlin\n */\npublic external abstract class Plugin
: ItemArrayLike<MimeType> {\n open val name: String\n open val description: String\n open val filename:
String\n override fun item(index: Int): MimeType?\n fun namedItem(name: String):
MimeType?\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun Plugin.get(index: Int):
MimeType? = asDynamic()[index]\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun Plugin.get(name: String):
MimeType? = asDynamic()[name]\n\n/**\n * Exposes the JavaScript
[MimeType](https://developer.mozilla.org/en/docs/Web/API/MimeType) to Kotlin\n */\npublic external abstract
class MimeType {\n open val type: String\n open val description: String\n open val suffixes: String\n open
val enabledPlugin: Plugin\n}\n\n/**\n * Exposes the JavaScript
[ImageBitmap](https://developer.mozilla.org/en/docs/Web/API/ImageBitmap) to Kotlin\n */\npublic external
abstract class ImageBitmap : CanvasImageSource, TexImageSource {\n open val width: Int\n open val height:
Int\n fun close()\n}\n\npublic external interface ImageBitmapOptions {\n var imageOrientation:
ImageOrientation? /* = ImageOrientation.NONE */\n get() = definedExternally\n set(value) =
definedExternally\n var premultiplyAlpha: PremultiplyAlpha? /* = PremultiplyAlpha.DEFAULT */\n get() =
definedExternally\n set(value) = definedExternally\n var colorSpaceConversion: ColorSpaceConversion? /* =
ColorSpaceConversion.DEFAULT */\n get() = definedExternally\n set(value) = definedExternally\n var
resizeWidth: Int?\n get() = definedExternally\n set(value) = definedExternally\n var resizeHeight: Int?\n
get() = definedExternally\n set(value) = definedExternally\n var resizeQuality: ResizeQuality? /* =
ResizeQuality.LOW */\n get() = definedExternally\n set(value) =

```

```

definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun
ImageBitmapOptions(imageOrientation: ImageOrientation? = ImageOrientation.NONE, premultiplyAlpha:
PremultiplyAlpha? = PremultiplyAlpha.DEFAULT, colorSpaceConversion: ColorSpaceConversion? =
ColorSpaceConversion.DEFAULT, resizeMode: Int? = undefined, resizeMode: Int? = undefined, resizeMode:
ResizeQuality? = ResizeQuality.LOW): ImageBitmapOptions {\n    val o = js(\"({})\")\n    o[\"imageOrientation\"] =
imageOrientation\n    o[\"premultiplyAlpha\"] = premultiplyAlpha\n    o[\"colorSpaceConversion\"] =
colorSpaceConversion\n    o[\"resizeWidth\"] = resizeMode\n    o[\"resizeHeight\"] = resizeMode\n
o[\"resizeQuality\"] = resizeMode\n    return o\n}\n\n/**\n * Exposes the JavaScript
[MessageEvent](https://developer.mozilla.org/en/docs/Web/API/MessageEvent) to Kotlin\n *\npublic external open
class MessageEvent(type: String, eventInitDict: MessageEventInit = definedExternally) : Event {\n    open val data:
Any?\n    open val origin: String\n    open val lastEventId: String\n    open val source:
UnionMessagePortOrWindowProxy?\n    open val ports: Array<out MessagePort>\n    fun initMessageEvent(type:
String, bubbles: Boolean, cancelable: Boolean, data: Any?, origin: String, lastEventId: String, source:
UnionMessagePortOrWindowProxy?, ports: Array<MessagePort>)\n\n    companion object {\n        val NONE:
Short\n        val CAPTURING_PHASE: Short\n        val AT_TARGET: Short\n        val BUBBLING_PHASE:
Short\n    }\n}\n\npublic external interface MessageEventInit : EventInit {\n    var data: Any? /* = null */\n    get()
= definedExternally\n    set(value) = definedExternally\n    var origin: String? /* = \"\" */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var lastEventId: String? /* = \"\" */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var source: UnionMessagePortOrWindowProxy? /* =
null */\n    get() = definedExternally\n    set(value) = definedExternally\n    var ports: Array<MessagePort>? /*
= arrayOf() */\n    get() = definedExternally\n    set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun MessageEventInit(data: Any? = null,
origin: String? = \"\", lastEventId: String? = \"\", source: UnionMessagePortOrWindowProxy? = null, ports:
Array<MessagePort>? = arrayOf(), bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? =
false): MessageEventInit {\n    val o = js(\"({})\")\n    o[\"data\"] = data\n    o[\"origin\"] = origin\n
o[\"lastEventId\"] = lastEventId\n    o[\"source\"] = source\n    o[\"ports\"] = ports\n    o[\"bubbles\"] = bubbles\n
o[\"cancelable\"] = cancelable\n    o[\"composed\"] = composed\n    return o\n}\n\n/**\n * Exposes the JavaScript
[EventSource](https://developer.mozilla.org/en/docs/Web/API/EventSource) to Kotlin\n *\npublic external open
class EventSource(url: String, eventSourceInitDict: EventSourceInit = definedExternally) : EventTarget {\n    open
val url: String\n    open val withCredentials: Boolean\n    open val readyState: Short\n    var onopen: ((Event) ->
dynamic)?\n    var onmessage: ((MessageEvent) -> dynamic)?\n    var onerror: ((Event) -> dynamic)?\n    fun
close()\n\n    companion object {\n        val CONNECTING: Short\n        val OPEN: Short\n        val CLOSED:
Short\n    }\n}\n\npublic external interface EventSourceInit {\n    var withCredentials: Boolean? /* = false */\n
get() = definedExternally\n    set(value) = definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun EventSourceInit(withCredentials:
Boolean? = false): EventSourceInit {\n    val o = js(\"({})\")\n    o[\"withCredentials\"] = withCredentials\n    return
o\n}\n\n/**\n * Exposes the JavaScript [WebSocket](https://developer.mozilla.org/en/docs/Web/API/WebSocket) to
Kotlin\n *\npublic external open class WebSocket(url: String, protocols: dynamic = definedExternally) :
EventTarget {\n    open val url: String\n    open val readyState: Short\n    open val bufferedAmount: Number\n    var
onopen: ((Event) -> dynamic)?\n    var onerror: ((Event) -> dynamic)?\n    var onclose: ((Event) -> dynamic)?\n
open val extensions: String\n    open val protocol: String\n    var onmessage: ((MessageEvent) -> dynamic)?\n    var
binaryType: BinaryType\n    fun close(code: Short = definedExternally, reason: String = definedExternally)\n    fun
send(data: String)\n    fun send(data: Blob)\n    fun send(data: ArrayBuffer)\n    fun send(data:
ArrayBufferView)\n\n    companion object {\n        val CONNECTING: Short\n        val OPEN: Short\n        val
CLOSING: Short\n        val CLOSED: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[CloseEvent](https://developer.mozilla.org/en/docs/Web/API/CloseEvent) to Kotlin\n *\npublic external open class

```

```

CloseEvent(type: String, eventInitDict: CloseEventInit = definedExternally) : Event {
    open val wasClean: Boolean
    open val code: Short
    open val reason: String
    companion object {
        val NONE: Short
        val CAPTURING_PHASE: Short
        val AT_TARGET: Short
        val BUBBLING_PHASE: Short
    }
}

public external interface CloseEventInit : EventInit {
    var wasClean: Boolean? /* = false */
    get() = definedExternally
    set(value) = definedExternally
    var code: Short? /* = 0 */
    get() = definedExternally
    set(value) = definedExternally
    var reason: String? /* = "" */
    get() = definedExternally
    set(value) = definedExternally
}

@Suppress("INVISIBLE_REFERENCE", "INVISIBLE_MEMBER")
@kotlin.internal.InlineOnly
public inline fun CloseEventInit(wasClean: Boolean? = false, code: Short? = 0, reason: String? = "", bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false): CloseEventInit {
    val o = js("{}")
    o["wasClean"] = wasClean
    o["code"] = code
    o["reason"] = reason
    o["bubbles"] = bubbles
    o["cancelable"] = cancelable
    o["composed"] = composed
    return o
}

/** Exposes the JavaScript [MessageChannel](https://developer.mozilla.org/en/docs/Web/API/MessageChannel) to Kotlin */
public external open class MessageChannel {
    open val port1: MessagePort
    open val port2: MessagePort
}

/** Exposes the JavaScript [MessagePort](https://developer.mozilla.org/en/docs/Web/API/MessagePort) to Kotlin */
public external abstract class MessagePort : EventTarget, UnionMessagePortOrWindowProxy, UnionMessagePortOrServiceWorker, UnionClientOrMessagePortOrServiceWorker {
    open var onmessage: ((MessageEvent) -> dynamic)?
    fun postMessage(message: Any?, transfer: Array<dynamic> = definedExternally)
    fun start()
    fun close()
}

/** Exposes the JavaScript [BroadcastChannel](https://developer.mozilla.org/en/docs/Web/API/BroadcastChannel) to Kotlin */
public external open class BroadcastChannel(name: String) : EventTarget {
    open val name: String
    var onmessage: ((MessageEvent) -> dynamic)?
    fun postMessage(message: Any?)
    fun close()
}

/** Exposes the JavaScript [WorkerGlobalScope](https://developer.mozilla.org/en/docs/Web/API/WorkerGlobalScope) to Kotlin */
public external abstract class WorkerGlobalScope : EventTarget, WindowOrWorkerGlobalScope, GlobalPerformance {
    open val self: WorkerGlobalScope
    open val location: WorkerLocation
    open val navigator: WorkerNavigator
    open var onerror: ((dynamic, String, Int, Int, Any?) -> dynamic)?
    open var onlanguagechange: ((Event) -> dynamic)?
    open var onoffline: ((Event) -> dynamic)?
    open var ononline: ((Event) -> dynamic)?
    open var onrejectionhandled: ((Event) -> dynamic)?
    open var onunhandledrejection: ((PromiseRejectionEvent) -> dynamic)?
    fun importScripts(vararg urls: String)
}

/** Exposes the JavaScript [DedicatedWorkerGlobalScope](https://developer.mozilla.org/en/docs/Web/API/DedicatedWorkerGlobalScope) to Kotlin */
public external abstract class DedicatedWorkerGlobalScope : WorkerGlobalScope {
    open var onmessage: ((MessageEvent) -> dynamic)?
    fun postMessage(message: Any?, transfer: Array<dynamic> = definedExternally)
    fun close()
}

/** Exposes the JavaScript [SharedWorkerGlobalScope](https://developer.mozilla.org/en/docs/Web/API/SharedWorkerGlobalScope) to Kotlin */
public external abstract class SharedWorkerGlobalScope : WorkerGlobalScope {
    open val name: String
    open val applicationCache: ApplicationCache
    open var onconnect: ((Event) -> dynamic)?
    fun close()
}

/** Exposes the JavaScript [AbstractWorker](https://developer.mozilla.org/en/docs/Web/API/AbstractWorker) to Kotlin */
public external interface AbstractWorker {
    var onerror: ((Event) -> dynamic)?
    get() = definedExternally
    set(value) = definedExternally
}

/** Exposes the JavaScript [Worker](https://developer.mozilla.org/en/docs/Web/API/Worker) to Kotlin */
public external open class Worker(scriptURL: String, options: WorkerOptions = definedExternally) : EventTarget, AbstractWorker {
    var onmessage: ((MessageEvent) -> dynamic)?
    override var onerror: ((Event) -> dynamic)?
    fun terminate()
    fun postMessage(message: Any?, transfer: Array<dynamic> = definedExternally)
}

public external interface WorkerOptions {
    var type: WorkerType? /* = WorkerType.CLASSIC */
    get() = definedExternally
    set(value) = definedExternally
    var credentials: RequestCredentials? /* = RequestCredentials.OMIT */
    get() = definedExternally
    set(value) = definedExternally
}

@Suppress("INVISIBLE_REFERENCE",

```

```

\@kotlin.internal.InlineOnly\npublic inline fun WorkerOptions(type: WorkerType? =
WorkerType.CLASSIC, credentials: RequestCredentials? = RequestCredentials.OMIT): WorkerOptions {
val o = js("{}")\n o["type"] = type\n o["credentials"] = credentials\n return o}\n\n/**\n * Exposes the
JavaScript [SharedWorker](https://developer.mozilla.org/en/docs/Web/API/SharedWorker) to Kotlin\n *\npublic
external open class SharedWorker(scriptURL: String, name: String = definedExternally, options: WorkerOptions =
definedExternally) : EventTarget, AbstractWorker {\n open val port: MessagePort\n override var onerror:
((Event) -> dynamic)?\n}\n\n/**\n * Exposes the JavaScript
[NavigatorConcurrentHardware](https://developer.mozilla.org/en/docs/Web/API/NavigatorConcurrentHardware) to
Kotlin\n *\npublic external interface NavigatorConcurrentHardware {\n val hardwareConcurrency:
Number\n}\n\n/**\n * Exposes the JavaScript
[WorkerNavigator](https://developer.mozilla.org/en/docs/Web/API/WorkerNavigator) to Kotlin\n *\npublic
external abstract class WorkerNavigator : NavigatorID, NavigatorLanguage, NavigatorOnLine,
NavigatorConcurrentHardware {\n open val serviceWorker: ServiceWorkerContainer\n}\n\n/**\n * Exposes the
JavaScript [WorkerLocation](https://developer.mozilla.org/en/docs/Web/API/WorkerLocation) to Kotlin\n
*\npublic external abstract class WorkerLocation {\n open val href: String\n open val origin: String\n open val
protocol: String\n open val host: String\n open val hostname: String\n open val port: String\n open val
pathname: String\n open val search: String\n open val hash: String\n}\n\n/**\n * Exposes the JavaScript
[Storage](https://developer.mozilla.org/en/docs/Web/API/Storage) to Kotlin\n *\npublic external abstract class
Storage {\n open val length: Int\n fun key(index: Int): String?\n fun removeItem(key: String)\n fun clear()\n
fun getItem(key: String): String?\n fun setItem(key: String, value:
String)\n}\n\n@Suppress("\@kotlin.internal.InlineOnly\npublic inline operator fun Storage.get(key: String):
String? = asDynamic()[key]\n\n@Suppress("\@kotlin.internal.InlineOnly\npublic inline operator fun Storage.set(key: String, value:
String) { asDynamic()[key] = value }\n\n/**\n * Exposes the JavaScript
[WindowSessionStorage](https://developer.mozilla.org/en/docs/Web/API/WindowSessionStorage) to Kotlin\n
*\npublic external interface WindowSessionStorage {\n val sessionStorage: Storage\n}\n\n/**\n * Exposes the
JavaScript [WindowLocalStorage](https://developer.mozilla.org/en/docs/Web/API/WindowLocalStorage) to
Kotlin\n *\npublic external interface WindowLocalStorage {\n val localStorage: Storage\n}\n\n/**\n * Exposes
the JavaScript [StorageEvent](https://developer.mozilla.org/en/docs/Web/API/StorageEvent) to Kotlin\n *\npublic
external open class StorageEvent(type: String, eventInitDict: StorageEventInit = definedExternally) : Event {\n
open val key: String?\n open val oldValue: String?\n open val newValue: String?\n open val url: String\n
open val storageArea: Storage?\n\n companion object {\n val NONE: Short\n val CAPTURING_PHASE:
Short\n val AT_TARGET: Short\n val BUBBLING_PHASE: Short\n }\n}\n\npublic external interface
StorageEventInit : EventInit {\n var key: String? /* = null */\n get() = definedExternally\n set(value) =
definedExternally\n var oldValue: String? /* = null */\n get() = definedExternally\n set(value) =
definedExternally\n var newValue: String? /* = null */\n get() = definedExternally\n set(value) =
definedExternally\n var url: String? /* = "" */\n get() = definedExternally\n set(value) =
definedExternally\n var storageArea: Storage? /* = null */\n get() = definedExternally\n set(value) =
definedExternally\n}\n\n@Suppress("\@kotlin.internal.InlineOnly\npublic inline fun StorageEventInit(key: String? = null,
oldValue: String? = null, newValue: String? = null, url: String? = "", storageArea: Storage? = null, bubbles:
Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false): StorageEventInit {\n val o =
js("{}")\n o["key"] = key\n o["oldValue"] = oldValue\n o["newValue"] = newValue\n o["url"] =
url\n o["storageArea"] = storageArea\n o["bubbles"] = bubbles\n o["cancelable"] = cancelable\n
o["composed"] = composed\n return o}\n}\n\npublic external abstract class HTMLElement :
HTMLElement {\n open var align: String\n open var alt: String\n open var archive: String\n open var code:
String\n open var codeBase: String\n open var height: String\n open var hspace: Int\n open var name:

```

```

String\n open var _object: String\n open var vspace: Int\n open var width: String\n\n companion object {\n
    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val
    CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE:
    Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
    DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
    DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
    DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
    DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n }
}\n\n/**\n * Exposes the JavaScript
[HTMLMarqueeElement](https://developer.mozilla.org/en/docs/Web/API/HTMLMarqueeElement) to Kotlin\n
*\npublic external abstract class HTMLMarqueeElement : HTMLElement {\n open var behavior: String\n open
var bgColor: String\n open var direction: String\n open var height: String\n open var hspace: Int\n open var
loop: Int\n open var scrollAmount: Int\n open var scrollDelay: Int\n open var trueSpeed: Boolean\n open var
vspace: Int\n open var width: String\n open var onbounce: ((Event) -> dynamic)?\n open var onfinish: ((Event)
-> dynamic)?\n open var onstart: ((Event) -> dynamic)?\n fun start()\n fun stop()\n\n companion object {\n
    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val
    CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE:
    Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
    DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
    DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
    DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
    DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n }
}\n\n/**\n * Exposes the JavaScript
[HTMLFrameSetElement](https://developer.mozilla.org/en/docs/Web/API/HTMLFrameSetElement) to Kotlin\n
*\npublic external abstract class HTMLFrameSetElement : HTMLElement, WindowEventHandlers {\n open var
cols: String\n open var rows: String\n\n companion object {\n    val ELEMENT_NODE: Short\n    val
    ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
    ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
    PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
    DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
    DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
    DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
    DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n }
}\n\npublic external abstract class
HTMLFrameElement : HTMLElement {\n open var name: String\n open var scrolling: String\n open var src:
String\n open var frameBorder: String\n open var longDesc: String\n open var noResize: Boolean\n open val
contentDocument: Document?\n open val contentWindow: Window?\n open var marginHeight: String\n open
var marginWidth: String\n\n companion object {\n    val ELEMENT_NODE: Short\n    val
    ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
    ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
    PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
    DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
    DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
    DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n

```

```

    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\npublic external abstract class
HTMLDirectoryElement : HTMLElement {\n    open var compact: Boolean\n\n    companion object {\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[HTMLFontElement](https://developer.mozilla.org/en/docs/Web/API/HTMLFontElement) to Kotlin\n *\npublic
external abstract class HTMLFontElement : HTMLElement {\n    open var color: String\n    open var face: String\n
    open var size: String\n\n    companion object {\n        val ELEMENT_NODE: Short\n        val
ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val
ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val
PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\npublic external interface External
{\n    fun AddSearchProvider()\n    fun IsSearchProviderInstalled()\n}\n\npublic external interface EventInit {\n
    var bubbles: Boolean? /* = false */\n        get() = definedExternally\n        set(value) = definedExternally\n
    var cancelable: Boolean? /* = false */\n        get() = definedExternally\n        set(value) = definedExternally\n
    var composed: Boolean? /* = false */\n        get() = definedExternally\n        set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\n\npublic inline fun EventInit(bubbles: Boolean? = false,
cancelable: Boolean? = false, composed: Boolean? = false): EventInit {\n    val o = js(\"({})\")\n    o[\"bubbles\"] =
bubbles\n    o[\"cancelable\"] = cancelable\n    o[\"composed\"] = composed\n    return o\n}\n\n/**\n * Exposes the
JavaScript [CustomEvent](https://developer.mozilla.org/en/docs/Web/API/CustomEvent) to Kotlin\n *\npublic
external open class CustomEvent(type: String, eventInitDict: CustomEventInit = definedExternally) : Event {\n
    open val detail: Any?\n    fun initCustomEvent(type: String, bubbles: Boolean, cancelable: Boolean, detail:
Any?)\n\n    companion object {\n        val NONE: Short\n        val CAPTURING_PHASE: Short\n        val
AT_TARGET: Short\n        val BUBBLING_PHASE: Short\n    }\n}\n\npublic external interface CustomEventInit :
EventInit {\n    var detail: Any? /* = null */\n        get() = definedExternally\n        set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\n\npublic inline fun CustomEventInit(detail: Any? = null,
bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false): CustomEventInit {\n    val o
= js(\"({})\")\n    o[\"detail\"] = detail\n    o[\"bubbles\"] = bubbles\n    o[\"cancelable\"] = cancelable\n
o[\"composed\"] = composed\n    return o\n}\n\npublic external interface EventListenerOptions {\n    var capture:
Boolean? /* = false */\n        get() = definedExternally\n        set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\n\npublic inline fun EventListenerOptions(capture:
Boolean? = false): EventListenerOptions {\n    val o = js(\"({})\")\n    o[\"capture\"] = capture\n    return
o\n}\n\npublic external interface AddEventListenerOptions : EventListenerOptions {\n    var passive: Boolean? /* =

```

```

false */\n    get() = definedExternally\n    set(value) = definedExternally\n    var once: Boolean? /* = false */\n    get() = definedExternally\n    set(value) = definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",  
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun AddEventListenerOptions(passive:  
Boolean? = false, once: Boolean? = false, capture: Boolean? = false): AddEventListenerOptions {\n    val o =  
js("{}")\n    o["passive"] = passive\n    o["once"] = once\n    o["capture"] = capture\n    return o\n}\n\npublic  
external interface NonElementParentNode {\n    fun getElementById(elementId: String): Element?\n}\n\n/**\n *  
Exposes the JavaScript  
[DocumentOrShadowRoot](https://developer.mozilla.org/en/docs/Web/API/DocumentOrShadowRoot) to Kotlin\n */\npublic external interface DocumentOrShadowRoot {\n    val fullscreenElement: Element?\n    get() =  
definedExternally\n}\n\n/**\n * Exposes the JavaScript  
[ParentNode](https://developer.mozilla.org/en/docs/Web/API/ParentNode) to Kotlin\n */\npublic external interface  
ParentNode {\n    val children: HTMLCollection\n    val firstElementChild: Element?\n    get() =  
definedExternally\n    val lastElementChild: Element?\n    get() = definedExternally\n    val childElementCount:  
Int\n    fun prepend(vararg nodes: dynamic)\n    fun append(vararg nodes: dynamic)\n    fun querySelector(selectors:  
String): Element?\n    fun querySelectorAll(selectors: String): NodeList\n}\n\n/**\n * Exposes the JavaScript  
[NonDocumentTypeChildNode](https://developer.mozilla.org/en/docs/Web/API/NonDocumentTypeChildNode) to  
Kotlin\n */\npublic external interface NonDocumentTypeChildNode {\n    val previousElementSibling: Element?\n    get() = definedExternally\n    val nextElementSibling: Element?\n    get() = definedExternally\n}\n\n/**\n *  
Exposes the JavaScript [ChildNode](https://developer.mozilla.org/en/docs/Web/API/ChildNode) to Kotlin\n */\npublic external interface ChildNode {\n    fun before(vararg nodes: dynamic)\n    fun after(vararg nodes:  
dynamic)\n    fun replaceWith(vararg nodes: dynamic)\n    fun remove()\n}\n\n/**\n * Exposes the JavaScript  
[Slotable](https://developer.mozilla.org/en/docs/Web/API/Slotable) to Kotlin\n */\npublic external interface Slotable  
{\n    val assignedSlot: HTMLSlotElement?\n    get() = definedExternally\n}\n\n/**\n * Exposes the JavaScript  
[NodeList](https://developer.mozilla.org/en/docs/Web/API/NodeList) to Kotlin\n */\npublic external abstract class  
NodeList : ItemArrayLike<Node> {\n    override fun item(index: Int):  
Node?\n}\n\n@Suppress("INVISIBLE_REFERENCE",  
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun NodeList.get(index: Int):  
Node? = asDynamic()[index]\n\n/**\n * Exposes the JavaScript  
[HTMLCollection](https://developer.mozilla.org/en/docs/Web/API/HTMLCollection) to Kotlin\n */\npublic  
external abstract class HTMLCollection : ItemArrayLike<Element>, UnionElementOrHTMLCollection {\n    override fun item(index: Int): Element?\n    fun namedItem(name: String):  
Element?\n}\n\n@Suppress("INVISIBLE_REFERENCE",  
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun HTMLCollection.get(index:  
Int): Element? = asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE",  
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun HTMLCollection.get(name:  
String): Element? = asDynamic()[name]\n\n/**\n * Exposes the JavaScript  
[MutationObserver](https://developer.mozilla.org/en/docs/Web/API/MutationObserver) to Kotlin\n */\npublic  
external open class MutationObserver(callback: (Array<MutationRecord>, MutationObserver) -> Unit) {\n    fun  
observe(target: Node, options: MutationObserverInit = definedExternally)\n    fun disconnect()\n    fun  
takeRecords(): Array<MutationRecord>\n}\n\n/**\n * Exposes the JavaScript  
[MutationObserverInit](https://developer.mozilla.org/en/docs/Web/API/MutationObserverInit) to Kotlin\n */\npublic external interface MutationObserverInit {\n    var childList: Boolean? /* = false */\n    get() =  
definedExternally\n    set(value) = definedExternally\n    var attributes: Boolean?\n    get() =  
definedExternally\n    set(value) = definedExternally\n    var characterData: Boolean?\n    get() =  
definedExternally\n    set(value) = definedExternally\n    var subtree: Boolean? /* = false */\n    get() =  
definedExternally\n    set(value) = definedExternally\n    var attributeOldValue: Boolean?\n    get() =  
definedExternally\n    set(value) = definedExternally\n    var characterDataOldValue: Boolean?\n    get() =  
definedExternally\n    set(value) = definedExternally\n    var attributeFilter: Array<String>?\n    get() =  
definedExternally\n    set(value) = definedExternally\n}

```

```

definedExternally\n    set(value) = definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun MutationObserverInit(childList:
Boolean? = false, attributes: Boolean? = undefined, characterData: Boolean? = undefined, subtree: Boolean? = false,
attributeOldValue: Boolean? = undefined, characterDataOldValue: Boolean? = undefined, attributeFilter:
Array<String>? = undefined): MutationObserverInit {\n    val o = js(\"({})\")\n    o[\"childList\"] = childList\n    o[\"attributes\"] = attributes\n    o[\"characterData\"] = characterData\n    o[\"subtree\"] = subtree\n    o[\"attributeOldValue\"] = attributeOldValue\n    o[\"characterDataOldValue\"] = characterDataOldValue\n    o[\"attributeFilter\"] = attributeFilter\n    return o\n}\n\n/**\n * Exposes the JavaScript
[MutationRecord](https://developer.mozilla.org/en/docs/Web/API/MutationRecord) to Kotlin\n */\npublic external
abstract class MutationRecord {\n    open val type: String\n    open val target: Node\n    open val addedNodes:
NodeList\n    open val removedNodes: NodeList\n    open val previousSibling: Node?\n    open val nextSibling:
Node?\n    open val attributeName: String?\n    open val attributeNamespace: String?\n    open val oldValue:
String?\n}\n\n/**\n * Exposes the JavaScript [Node](https://developer.mozilla.org/en/docs/Web/API/Node) to
Kotlin\n */\npublic external abstract class Node : EventTarget {\n    open val nodeType: Short\n    open val
nodeName: String\n    open val baseURI: String\n    open val isConnected: Boolean\n    open val ownerDocument:
Document?\n    open val parentNode: Node?\n    open val parentElement: Element?\n    open val childNodes:
NodeList\n    open val firstChild: Node?\n    open val lastChild: Node?\n    open val previousSibling: Node?\n
open val nextSibling: Node?\n    open var nodeValue: String?\n    open var textContent: String?\n    fun
getRootNode(options: GetRootNodeOptions = definedExternally): Node\n    fun hasChildNodes(): Boolean\n    fun
normalize()\n    fun cloneNode(deep: Boolean = definedExternally): Node\n    fun isEqualNode(otherNode: Node?):
Boolean\n    fun isSameNode(otherNode: Node?): Boolean\n    fun compareDocumentPosition(other: Node): Short\n
    fun contains(other: Node?): Boolean\n    fun lookupPrefix(namespace: String?): String?\n    fun
lookupNamespaceURI(prefix: String?): String?\n    fun isDefaultNamespace(namespace: String?): Boolean\n    fun
insertBefore(node: Node, child: Node?): Node\n    fun appendChild(node: Node): Node\n    fun replaceChild(node:
Node, child: Node): Node\n    fun removeChild(child: Node): Node\n\n    companion object {\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\npublic external interface
GetRootNodeOptions {\n    var composed: Boolean? /* = false */\n    get() = definedExternally\n    set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun GetRootNodeOptions(composed:
Boolean? = false): GetRootNodeOptions {\n    val o = js(\"({})\")\n    o[\"composed\"] = composed\n    return
o\n}\n\n/**\n * Exposes the JavaScript [Document](https://developer.mozilla.org/en/docs/Web/API/Document) to
Kotlin\n */\npublic external open class Document : Node, GlobalEventHandlers,
DocumentAndElementEventHandlers, NonElementParentNode, DocumentOrShadowRoot, ParentNode,
GeometryUtils {\n    open val implementation: DOMImplementation\n    open val URL: String\n    open val
documentURI: String\n    open val origin: String\n    open val compatMode: String\n    open val characterSet:
String\n    open val charset: String\n    open val inputEncoding: String\n    open val contentType: String\n    open val
doctype: DocumentType?\n    open val documentElement: Element?\n    open val location: Location?\n    var
domain: String\n    open val referrer: String\n    var cookie: String\n    open val lastModified: String\n    open val
readyState: DocumentReadyState\n    var title: String\n    var dir: String\n    var body: HTMLElement?\n    open val
head: HTMLHeadElement?\n    open val images: HTMLCollection\n    open val embeds: HTMLCollection\n    open

```

```

val plugins: HTMLCollection\n open val links: HTMLCollection\n open val forms: HTMLCollection\n open
val scripts: HTMLCollection\n open val currentScript: HTMLOrSVGScriptElement?\n open val defaultView:
Window?\n open val activeElement: Element?\n var designMode: String\n var onreadystatechange: ((Event) ->
dynamic)?\n var fgColor: String\n var linkColor: String\n var vlinkColor: String\n var alinkColor: String\n
var bgColor: String\n open val anchors: HTMLCollection\n open val applets: HTMLCollection\n open val all:
HTMLAllCollection\n open val scrollingElement: Element?\n open val styleSheets: StyleSheetList\n open val
rootElement: SVGSVGElement?\n open val fullscreenEnabled: Boolean\n open val fullscreen: Boolean\n var
onfullscreenchange: ((Event) -> dynamic)?\n var onfullscreenerror: ((Event) -> dynamic)?\n override var
onabort: ((Event) -> dynamic)?\n override var onblur: ((FocusEvent) -> dynamic)?\n override var oncancel:
((Event) -> dynamic)?\n override var oncanplay: ((Event) -> dynamic)?\n override var oncanplaythrough:
((Event) -> dynamic)?\n override var onchange: ((Event) -> dynamic)?\n override var onclick: ((MouseEvent) ->
dynamic)?\n override var onclose: ((Event) -> dynamic)?\n override var oncontextmenu: ((MouseEvent) ->
dynamic)?\n override var oncuechange: ((Event) -> dynamic)?\n override var ondblclick: ((MouseEvent) ->
dynamic)?\n override var ondrag: ((DragEvent) -> dynamic)?\n override var ondragend: ((DragEvent) ->
dynamic)?\n override var ondragenter: ((DragEvent) -> dynamic)?\n override var ondragexit: ((DragEvent) ->
dynamic)?\n override var ondragleave: ((DragEvent) -> dynamic)?\n override var ondragover: ((DragEvent) ->
dynamic)?\n override var ondragstart: ((DragEvent) -> dynamic)?\n override var ondrop: ((DragEvent) ->
dynamic)?\n override var ondurationchange: ((Event) -> dynamic)?\n override var onemptied: ((Event) ->
dynamic)?\n override var onended: ((Event) -> dynamic)?\n override var onerror: ((dynamic, String, Int, Int,
Any?) -> dynamic)?\n override var onfocus: ((FocusEvent) -> dynamic)?\n override var oninput: ((InputEvent) ->
dynamic)?\n override var oninvalid: ((Event) -> dynamic)?\n override var onkeydown: ((KeyboardEvent) ->
dynamic)?\n override var onkeypress: ((KeyboardEvent) -> dynamic)?\n override var onkeyup:
((KeyboardEvent) -> dynamic)?\n override var onload: ((Event) -> dynamic)?\n override var onloadeddata:
((Event) -> dynamic)?\n override var onloadedmetadata: ((Event) -> dynamic)?\n override var onloadend:
((Event) -> dynamic)?\n override var onloadstart: ((ProgressEvent) -> dynamic)?\n override var onmousedown:
((MouseEvent) -> dynamic)?\n override var onmouseenter: ((MouseEvent) -> dynamic)?\n override var
onmouseleave: ((MouseEvent) -> dynamic)?\n override var onmousemove: ((MouseEvent) -> dynamic)?\n
override var onmouseout: ((MouseEvent) -> dynamic)?\n override var onmouseover: ((MouseEvent) ->
dynamic)?\n override var onmouseup: ((MouseEvent) -> dynamic)?\n override var onwheel: ((WheelEvent) ->
dynamic)?\n override var onpause: ((Event) -> dynamic)?\n override var onplay: ((Event) -> dynamic)?\n
override var onplaying: ((Event) -> dynamic)?\n override var onprogress: ((ProgressEvent) -> dynamic)?\n
override var onratechange: ((Event) -> dynamic)?\n override var onreset: ((Event) -> dynamic)?\n override var
onresize: ((Event) -> dynamic)?\n override var onscroll: ((Event) -> dynamic)?\n override var onseeked:
((Event) -> dynamic)?\n override var onseeking: ((Event) -> dynamic)?\n override var onselect: ((Event) ->
dynamic)?\n override var onshow: ((Event) -> dynamic)?\n override var onstalled: ((Event) -> dynamic)?\n
override var onsubmit: ((Event) -> dynamic)?\n override var onsuspend: ((Event) -> dynamic)?\n override var
ontimeupdate: ((Event) -> dynamic)?\n override var ontoggle: ((Event) -> dynamic)?\n override var
onvolumechange: ((Event) -> dynamic)?\n override var onwaiting: ((Event) -> dynamic)?\n override var
ongotpointercapture: ((PointerEvent) -> dynamic)?\n override var onlostpointercapture: ((PointerEvent) ->
dynamic)?\n override var onpointerdown: ((PointerEvent) -> dynamic)?\n override var onpointermove:
((PointerEvent) -> dynamic)?\n override var onpointerup: ((PointerEvent) -> dynamic)?\n override var
onpointercancel: ((PointerEvent) -> dynamic)?\n override var onpointerover: ((PointerEvent) -> dynamic)?\n
override var onpointerout: ((PointerEvent) -> dynamic)?\n override var onpointerenter: ((PointerEvent) ->
dynamic)?\n override var onpointerleave: ((PointerEvent) -> dynamic)?\n override var oncopy:
((ClipboardEvent) -> dynamic)?\n override var oncut: ((ClipboardEvent) -> dynamic)?\n override var onpaste:
((ClipboardEvent) -> dynamic)?\n override val fullscreenElement: Element?\n override val children:
HTMLCollection\n override val firstElementChild: Element?\n override val lastElementChild: Element?\n
override val childElementCount: Int\n fun getElementsByTagName(qualifiedName: String): HTMLCollection\n

```

```

fun getElementsByTagName(namespace: String?, localName: String): HTMLCollection\n fun
getElementsByClassName(classNames: String): HTMLCollection\n fun createElement(localName: String,
options: ElementCreationOptions = definedExternally): Element\n fun createElementNS(namespace: String?,
qualifiedName: String, options: ElementCreationOptions = definedExternally): Element\n fun
createDocumentFragment(): DocumentFragment\n fun createTextNode(data: String): Text\n fun
createCDATASection(data: String): CDATASection\n fun createComment(data: String): Comment\n fun
createProcessingInstruction(target: String, data: String): ProcessingInstruction\n fun importNode(node: Node,
deep: Boolean = definedExternally): Node\n fun adoptNode(node: Node): Node\n fun
createAttribute(localName: String): Attr\n fun createAttributeNS(namespace: String?, qualifiedName: String):
Attr\n fun createEvent(`interface`: String): Event\n fun createRange(): Range\n fun createNodeIterator(root:
Node, whatToShow: Int = definedExternally, filter: NodeFilter? = definedExternally): NodeIterator\n fun
createNodeIterator(root: Node, whatToShow: Int = definedExternally, filter: ((Node) -> Short)? =
definedExternally): NodeIterator\n fun createTreeWalker(root: Node, whatToShow: Int = definedExternally, filter:
NodeFilter? = definedExternally): TreeWalker\n fun createTreeWalker(root: Node, whatToShow: Int =
definedExternally, filter: ((Node) -> Short)? = definedExternally): TreeWalker\n fun
getElementsByName(elementName: String): NodeList\n fun open(type: String = definedExternally, replace:
String = definedExternally): Document\n fun open(url: String, name: String, features: String): Window\n fun
close()\n fun write(vararg text: String)\n fun writeln(vararg text: String)\n fun hasFocus(): Boolean\n fun
execCommand(commandId: String, showUI: Boolean = definedExternally, value: String = definedExternally):
Boolean\n fun queryCommandEnabled(commandId: String): Boolean\n fun
queryCommandIndeterm(commandId: String): Boolean\n fun queryCommandState(commandId: String):
Boolean\n fun queryCommandSupported(commandId: String): Boolean\n fun
queryCommandValue(commandId: String): String\n fun clear()\n fun captureEvents()\n fun releaseEvents()\n
fun elementFromPoint(x: Double, y: Double): Element?\n fun elementsFromPoint(x: Double, y: Double):
Array<Element>\n fun caretPositionFromPoint(x: Double, y: Double): CaretPosition?\n fun createTouch(view:
Window, target: EventTarget, identifier: Int, pageX: Int, pageY: Int, screenX: Int, screenY: Int): Touch\n fun
createTouchList(vararg touches: Touch): TouchList\n fun exitFullscreen(): Promise<Unit>\n override fun
getElementById(elementId: String): Element?\n override fun prepend(vararg nodes: dynamic)\n override fun
append(vararg nodes: dynamic)\n override fun querySelector(selectors: String): Element?\n override fun
querySelectorAll(selectors: String): NodeList\n override fun getBoxQuads(options: BoxQuadOptions /* =
definedExternally */): Array<DOMQuad>\n override fun convertQuadFromNode(quad: dynamic, from: dynamic,
options: ConvertCoordinateOptions /* = definedExternally */): DOMQuad\n override fun
convertRectFromNode(rect: DOMRectReadOnly, from: dynamic, options: ConvertCoordinateOptions /* =
definedExternally */): DOMQuad\n override fun convertPointFromNode(point: DOMPointInit, from: dynamic,
options: ConvertCoordinateOptions /* = definedExternally */): DOMPoint\n\n companion object {\n val
ELEMENT_NODE: Short\n val ATTRIBUTE_NODE: Short\n val TEXT_NODE: Short\n val
CDATA_SECTION_NODE: Short\n val ENTITY_REFERENCE_NODE: Short\n val ENTITY_NODE:
Short\n val PROCESSING_INSTRUCTION_NODE: Short\n val COMMENT_NODE: Short\n val
DOCUMENT_NODE: Short\n val DOCUMENT_TYPE_NODE: Short\n val
DOCUMENT_FRAGMENT_NODE: Short\n val NOTATION_NODE: Short\n val
DOCUMENT_POSITION_DISCONNECTED: Short\n val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n
}\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun Document.get(name: String):
dynamic = asDynamic()[name]\n\n/**\n * Exposes the JavaScript
[XMLDocument](https://developer.mozilla.org/en/docs/Web/API/XMLDocument) to Kotlin\n */\npublic external

```

```

open class XMLDocument : Document {
    companion object {
        val ELEMENT_NODE: Short
        val ATTRIBUTE_NODE: Short
        val TEXT_NODE: Short
        val CDATA_SECTION_NODE: Short
        val ENTITY_REFERENCE_NODE: Short
        val ENTITY_NODE: Short
        val PROCESSING_INSTRUCTION_NODE: Short
        val COMMENT_NODE: Short
        val DOCUMENT_NODE: Short
        val DOCUMENT_TYPE_NODE: Short
        val DOCUMENT_FRAGMENT_NODE: Short
        val NOTATION_NODE: Short
        val DOCUMENT_POSITION_DISCONNECTED: Short
        val DOCUMENT_POSITION_PRECEDING: Short
        val DOCUMENT_POSITION_FOLLOWING: Short
        val DOCUMENT_POSITION_CONTAINS: Short
        val DOCUMENT_POSITION_CONTAINED_BY: Short
    }
}

DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short }

public external interface
ElementCreationOptions {
    var `is`: String?
    get() = definedExternally
    set(value) =
definedExternally
}

@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")
@kotlin.internal.InlineOnly
public inline fun ElementCreationOptions(`is`: String?
= undefined): ElementCreationOptions {
    val o = js("{}")
    o["is"] = `is`
    return o
}

/**
 * Exposes the JavaScript
 [DOMImplementation](https://developer.mozilla.org/en/docs/Web/API/DOMImplementation) to Kotlin
 */
public external abstract class DOMImplementation {
    fun createDocumentType(qualifiedName: String,
publicId: String, systemId: String): DocumentType
    fun createDocument(namespace: String?, qualifiedName:
String, doctype: DocumentType? = definedExternally): XMLDocument
    fun createHTMLDocument(title: String
= definedExternally): Document
    fun hasFeature(): Boolean
}

/**
 * Exposes the JavaScript
 [DocumentType](https://developer.mozilla.org/en/docs/Web/API/DocumentType) to Kotlin
 */
public external
abstract class DocumentType : Node, ChildNode {
    open val name: String
    open val publicId: String
    open
val systemId: String
    companion object {
        val ELEMENT_NODE: Short
        val ATTRIBUTE_NODE:
Short
        val TEXT_NODE: Short
        val CDATA_SECTION_NODE: Short
        val
ENTITY_REFERENCE_NODE: Short
        val ENTITY_NODE: Short
        val
PROCESSING_INSTRUCTION_NODE: Short
        val COMMENT_NODE: Short
        val
DOCUMENT_NODE: Short
        val DOCUMENT_TYPE_NODE: Short
        val
DOCUMENT_FRAGMENT_NODE: Short
        val NOTATION_NODE: Short
        val
DOCUMENT_POSITION_DISCONNECTED: Short
        val DOCUMENT_POSITION_PRECEDING: Short
        val
DOCUMENT_POSITION_FOLLOWING: Short
        val DOCUMENT_POSITION_CONTAINS: Short
        val
DOCUMENT_POSITION_CONTAINED_BY: Short
        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short
    }
}

/**
 * Exposes the JavaScript
 [DocumentFragment](https://developer.mozilla.org/en/docs/Web/API/DocumentFragment) to Kotlin
 */
public
external open class DocumentFragment : Node, NonElementParentNode, ParentNode {
    override val children:
HTMLCollection
    override val firstElementChild: Element?
    override val lastElementChild: Element?
    override val childElementCount: Int
    override fun getElementById(elementId: String): Element?
    override fun
prepend(vararg nodes: dynamic)
    override fun append(vararg nodes: dynamic)
    override fun
querySelector(selectors: String): Element?
    override fun querySelectorAll(selectors: String): NodeList
    companion object {
        val ELEMENT_NODE: Short
        val ATTRIBUTE_NODE: Short
        val
TEXT_NODE: Short
        val CDATA_SECTION_NODE: Short
        val ENTITY_REFERENCE_NODE:
Short
        val ENTITY_NODE: Short
        val PROCESSING_INSTRUCTION_NODE: Short
        val
COMMENT_NODE: Short
        val DOCUMENT_NODE: Short
        val DOCUMENT_TYPE_NODE: Short
        val
DOCUMENT_FRAGMENT_NODE: Short
        val NOTATION_NODE: Short
        val
DOCUMENT_POSITION_DISCONNECTED: Short
        val DOCUMENT_POSITION_PRECEDING: Short
        val
DOCUMENT_POSITION_FOLLOWING: Short
        val DOCUMENT_POSITION_CONTAINS: Short
        val
DOCUMENT_POSITION_CONTAINED_BY: Short
        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short
    }
}

/**
 * Exposes the JavaScript
 [ShadowRoot](https://developer.mozilla.org/en/docs/Web/API/ShadowRoot) to Kotlin
 */
public external open

```

```

class ShadowRoot : DocumentFragment, DocumentOrShadowRoot {
    open val mode: ShadowRootMode
    open val host: Element
    override val fullscreenElement: Element?
    companion object {
        val ELEMENT_NODE: Short
        val ATTRIBUTE_NODE: Short
        val TEXT_NODE: Short
        val CDATA_SECTION_NODE: Short
        val ENTITY_REFERENCE_NODE: Short
        val ENTITY_NODE: Short
        val PROCESSING_INSTRUCTION_NODE: Short
        val COMMENT_NODE: Short
        val DOCUMENT_NODE: Short
        val DOCUMENT_TYPE_NODE: Short
        val DOCUMENT_FRAGMENT_NODE: Short
        val NOTATION_NODE: Short
        val DOCUMENT_POSITION_DISCONNECTED: Short
        val DOCUMENT_POSITION_PRECEDING: Short
        val DOCUMENT_POSITION_FOLLOWING: Short
        val DOCUMENT_POSITION_CONTAINS: Short
        val DOCUMENT_POSITION_CONTAINED_BY: Short
        val DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short
    }
}

/** Exposes the JavaScript [Element](https://developer.mozilla.org/en/docs/Web/API/Element) to Kotlin
 *
 * \npublic external abstract class Element : Node, ParentNode, NonDocumentTypeChildNode, ChildNode, Slotable, GeometryUtils, UnionElementOrHTMLCollection, UnionElementOrRadioNodeList, UnionElementOrMouseEvent, UnionElementOrProcessingInstruction {
    open val namespaceURI: String?
    open val prefix: String?
    open val localName: String
    open val tagName: String
    open var id: String
    open var className: String
    open val classList: DOMTokenList
    open var slot: String
    open val attributes: NamedNodeMap
    open val shadowRoot: ShadowRoot?
    open var scrollTop: Double
    open var scrollLeft: Double
    open val scrollWidth: Int
    open val scrollHeight: Int
    open val clientTop: Int
    open val clientLeft: Int
    open val clientWidth: Int
    open val clientHeight: Int
    open var innerHTML: String
    open var outerHTML: String
    fun hasAttributes(): Boolean
    fun getAttributeNames(): Array<String>
    fun getAttribute(qualifiedName: String): String?
    fun getAttributeNS(namespace: String?, localName: String): String?
    fun setAttribute(qualifiedName: String, value: String)
    fun setAttributeNS(namespace: String?, qualifiedName: String, value: String)
    fun removeAttribute(qualifiedName: String)
    fun removeAttributeNS(namespace: String?, localName: String)
    fun hasAttribute(qualifiedName: String): Boolean
    fun hasAttributeNS(namespace: String?, localName: String): Boolean
    fun getAttributeNode(qualifiedName: String): Attr?
    fun getAttributeNodeNS(namespace: String?, localName: String): Attr?
    fun setAttributeNode(attr: Attr): Attr?
    fun setAttributeNodeNS(attr: Attr): Attr?
    fun removeAttributeNode(attr: Attr): Attr
    fun attachShadow(init: ShadowRootInit): ShadowRoot
    fun closest(selectors: String): Element?
    fun matches(selectors: String): Boolean
    fun webkitMatchesSelector(selectors: String): Boolean
    fun getElementsByTagName(qualifiedName: String): HTMLCollection
    fun getElementsByTagNameNS(namespace: String?, localName: String): HTMLCollection
    fun getElementsByTagNameNS(classNames: String): HTMLCollection
    fun insertAdjacentElement(where: String, element: Element): Element?
    fun insertAdjacentText(where: String, data: String)
    fun getClientRects(): Array<DOMRect>
    fun getBoundingClientRect(): DOMRect
    fun scrollIntoView()
    fun scrollIntoView(arg: dynamic)
    fun scroll(options: ScrollToOptions = definedExternally)
    fun scroll(x: Double, y: Double)
    fun scrollTo(options: ScrollToOptions = definedExternally)
    fun scrollTo(x: Double, y: Double)
    fun scrollBy(options: ScrollToOptions = definedExternally)
    fun scrollBy(x: Double, y: Double)
    fun insertAdjacentHTML(position: String, text: String)
    fun setPointerCapture(pointerId: Int)
    fun releasePointerCapture(pointerId: Int)
    fun hasPointerCapture(pointerId: Int): Boolean
    fun requestFullscreen(): Promise<Unit>
}

companion object {
    val ELEMENT_NODE: Short
    val ATTRIBUTE_NODE: Short
    val TEXT_NODE: Short
    val CDATA_SECTION_NODE: Short
    val ENTITY_REFERENCE_NODE: Short
    val ENTITY_NODE: Short
    val PROCESSING_INSTRUCTION_NODE: Short
    val COMMENT_NODE: Short
    val DOCUMENT_NODE: Short
    val DOCUMENT_TYPE_NODE: Short
    val DOCUMENT_FRAGMENT_NODE: Short
    val NOTATION_NODE: Short
    val DOCUMENT_POSITION_DISCONNECTED: Short
    val DOCUMENT_POSITION_PRECEDING: Short
    val DOCUMENT_POSITION_FOLLOWING: Short
    val DOCUMENT_POSITION_CONTAINS: Short
}

```

```

    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\npublic external interface
ShadowRootInit {\n    var mode: ShadowRootMode?\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun ShadowRootInit(mode:
ShadowRootMode?): ShadowRootInit {\n    val o = js("{}")\n    o["mode"] = mode\n    return o\n}\n\n/**\n *
Exposes the JavaScript [NamedNodeMap](https://developer.mozilla.org/en/docs/Web/API/NamedNodeMap) to
Kotlin\n */\npublic external abstract class NamedNodeMap : ItemArrayLike<Attr> {\n    fun
getNamedItemNS(namespace: String?, localName: String): Attr?\n    fun setNamedItem(attr: Attr): Attr?\n    fun
setNamedItemNS(attr: Attr): Attr?\n    fun removeNamedItem(qualifiedName: String): Attr?\n    fun
removeNamedItemNS(namespace: String?, localName: String): Attr?\n    override fun item(index: Int): Attr?\n    fun
getNamedItem(qualifiedName: String): Attr?\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun NamedNodeMap.get(index:
Int): Attr? = asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun
NamedNodeMap.get(qualifiedName: String): Attr? = asDynamic()[qualifiedName]\n\n/**\n * Exposes the
JavaScript [Attr](https://developer.mozilla.org/en/docs/Web/API/Attr) to Kotlin\n */\npublic external abstract class
Attr : Node {\n    open val namespaceURI: String?\n    open val prefix: String?\n    open val localName: String\n    open val name: String\n    open var value: String\n    open val ownerElement: Element?\n    open val specified:
Boolean\n\n    companion object {\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE:
Short\n        val ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n    }\n\n    DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[CharacterData](https://developer.mozilla.org/en/docs/Web/API/CharacterData) to Kotlin\n */\npublic external
abstract class CharacterData : Node, NonDocumentTypeChildNode, ChildNode {\n    open var data: String\n    open
val length: Int\n    fun substringData(offset: Int, count: Int): String\n    fun appendData(data: String)\n    fun
insertData(offset: Int, data: String)\n    fun deleteData(offset: Int, count: Int)\n    fun replaceData(offset: Int, count:
Int, data: String)\n\n    companion object {\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE:
Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n    }\n\n    DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[Text](https://developer.mozilla.org/en/docs/Web/API/Text) to Kotlin\n */\npublic external open class Text(data:
String = definedExternally) : CharacterData, Slotable, GeometryUtils {\n    open val wholeText: String\n    override
val assignedSlot: HTMLSlotElement?\n    override val previousElementSibling: Element?\n    override val
nextElementSibling: Element?\n    fun splitText(offset: Int): Text\n    override fun getBoxQuads(options:
BoxQuadOptions /* = definedExternally */): Array<DOMQuad>\n    override fun convertQuadFromNode(quad:
dynamic, from: dynamic, options: ConvertCoordinateOptions /* = definedExternally */): DOMQuad\n    override
fun convertRectFromNode(rect: DOMRectReadOnly, from: dynamic, options: ConvertCoordinateOptions /* =

```

```

definedExternally */): DOMQuad\n  override fun convertPointFromNode(point: DOMPointInit, from: dynamic,
options: ConvertCoordinateOptions /* = definedExternally */): DOMPoint\n  override fun before(vararg nodes:
dynamic)\n  override fun after(vararg nodes: dynamic)\n  override fun replaceWith(vararg nodes: dynamic)\n
override fun remove()\n\n  companion object {\n    val ELEMENT_NODE: Short\n    val
ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n\n  /**\n   * Exposes the JavaScript
[CDATASection](https://developer.mozilla.org/en/docs/Web/API/CDATASection) to Kotlin\n   */\n  public external
open class CDATASection : Text {\n    companion object {\n      val ELEMENT_NODE: Short\n      val
ATTRIBUTE_NODE: Short\n      val TEXT_NODE: Short\n      val CDATA_SECTION_NODE: Short\n      val
ENTITY_REFERENCE_NODE: Short\n      val ENTITY_NODE: Short\n      val
PROCESSING_INSTRUCTION_NODE: Short\n      val COMMENT_NODE: Short\n      val
DOCUMENT_NODE: Short\n      val DOCUMENT_TYPE_NODE: Short\n      val
DOCUMENT_FRAGMENT_NODE: Short\n      val NOTATION_NODE: Short\n      val
DOCUMENT_POSITION_DISCONNECTED: Short\n      val DOCUMENT_POSITION_PRECEDING: Short\n
      val DOCUMENT_POSITION_FOLLOWING: Short\n      val DOCUMENT_POSITION_CONTAINS: Short\n
      val DOCUMENT_POSITION_CONTAINED_BY: Short\n      val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n\n    /**\n     * Exposes the JavaScript
[ProcessingInstruction](https://developer.mozilla.org/en/docs/Web/API/ProcessingInstruction) to Kotlin\n     */\n    public external abstract class ProcessingInstruction : CharacterData, LinkStyle,
UnionElementOrProcessingInstruction {\n      open val target: String\n      companion object {\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n      }\n\n      /**\n       * Exposes the JavaScript
[Comment](https://developer.mozilla.org/en/docs/Web/API/Comment) to Kotlin\n       */\n      public external open class
Comment(data: String = definedExternally) : CharacterData {\n        override val previousElementSibling: Element?\n
        override val nextElementSibling: Element?\n        override fun before(vararg nodes: dynamic)\n        override fun
after(vararg nodes: dynamic)\n        override fun replaceWith(vararg nodes: dynamic)\n        override fun remove()\n
        companion object {\n          val ELEMENT_NODE: Short\n          val ATTRIBUTE_NODE: Short\n          val
TEXT_NODE: Short\n          val CDATA_SECTION_NODE: Short\n          val ENTITY_REFERENCE_NODE:
Short\n          val ENTITY_NODE: Short\n          val PROCESSING_INSTRUCTION_NODE: Short\n          val
COMMENT_NODE: Short\n          val DOCUMENT_NODE: Short\n          val DOCUMENT_TYPE_NODE: Short\n
          val DOCUMENT_FRAGMENT_NODE: Short\n          val NOTATION_NODE: Short\n          val
DOCUMENT_POSITION_DISCONNECTED: Short\n          val DOCUMENT_POSITION_PRECEDING: Short\n
          val DOCUMENT_POSITION_FOLLOWING: Short\n          val DOCUMENT_POSITION_CONTAINS: Short\n
          val DOCUMENT_POSITION_CONTAINED_BY: Short\n          val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n        }\n      }\n    }\n  }\n}

```



```

    override var x: Double\n    override var y: Double\n    override var z: Double\n    override var w:
Double\n}\n\n/**\n * Exposes the JavaScript
[DOMPointInit](https://developer.mozilla.org/en/docs/Web/API/DOMPointInit) to Kotlin\n */\npublic external
interface DOMPointInit {\n    var x: Double? /* = 0.0 */\n        get() = definedExternally\n        set(value) =
definedExternally\n    var y: Double? /* = 0.0 */\n        get() = definedExternally\n        set(value) =
definedExternally\n    var z: Double? /* = 0.0 */\n        get() = definedExternally\n        set(value) =
definedExternally\n    var w: Double? /* = 1.0 */\n        get() = definedExternally\n        set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun DOMPointInit(x: Double? = 0.0, y:
Double? = 0.0, z: Double? = 0.0, w: Double? = 1.0): DOMPointInit {\n    val o = js(\"({})\")\n    o[\"x\"] = x\n
o[\"y\"] = y\n    o[\"z\"] = z\n    o[\"w\"] = w\n    return o\n}\n\n/**\n * Exposes the JavaScript
[DOMRect](https://developer.mozilla.org/en/docs/Web/API/DOMRect) to Kotlin\n */\npublic external open class
DOMRect(x: Double = definedExternally, y: Double = definedExternally, width: Double = definedExternally,
height: Double = definedExternally) : DOMRectReadOnly {\n    override var x: Double\n        override var y: Double\n
        override var width: Double\n        override var height: Double\n}\n\n/**\n * Exposes the JavaScript
[DOMRectReadOnly](https://developer.mozilla.org/en/docs/Web/API/DOMRectReadOnly) to Kotlin\n */\npublic
external open class DOMRectReadOnly(x: Double, y: Double, width: Double, height: Double) {\n    open val x:
Double\n    open val y: Double\n    open val width: Double\n    open val height: Double\n    open val top: Double\n
    open val right: Double\n    open val bottom: Double\n    open val left: Double\n}\n\npublic external interface
DOMRectInit {\n    var x: Double? /* = 0.0 */\n        get() = definedExternally\n        set(value) =
definedExternally\n    var y: Double? /* = 0.0 */\n        get() = definedExternally\n        set(value) =
definedExternally\n    var width: Double? /* = 0.0 */\n        get() = definedExternally\n        set(value) =
definedExternally\n    var height: Double? /* = 0.0 */\n        get() = definedExternally\n        set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun DOMRectInit(x: Double? = 0.0, y:
Double? = 0.0, width: Double? = 0.0, height: Double? = 0.0): DOMRectInit {\n    val o = js(\"({})\")\n    o[\"x\"] =
x\n    o[\"y\"] = y\n    o[\"width\"] = width\n    o[\"height\"] = height\n    return o\n}\n\npublic external interface
DOMRectList : ItemArrayLike<DOMRect> {\n    override fun item(index: Int):
DOMRect?\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun DOMRectList.get(index: Int):
DOMRect? = asDynamic()[index]\n\n/**\n * Exposes the JavaScript
[DOMQuad](https://developer.mozilla.org/en/docs/Web/API/DOMQuad) to Kotlin\n */\npublic external open class
DOMQuad {\n    constructor(p1: DOMPointInit = definedExternally, p2: DOMPointInit = definedExternally, p3:
DOMPointInit = definedExternally, p4: DOMPointInit = definedExternally)\n    constructor(rect: DOMRectInit)\n
    open val p1: DOMPoint\n    open val p2: DOMPoint\n    open val p3: DOMPoint\n    open val p4: DOMPoint\n
    open val bounds: DOMRectReadOnly\n}\n\n/**\n * Exposes the JavaScript
[DOMMatrixReadOnly](https://developer.mozilla.org/en/docs/Web/API/DOMMatrixReadOnly) to Kotlin\n */\npublic
external open class DOMMatrixReadOnly(numberSequence: Array<Double>) {\n    open val a: Double\n    open val b:
Double\n    open val c: Double\n    open val d: Double\n    open val e: Double\n    open val f: Double\n    open
val m11: Double\n    open val m12: Double\n    open val m13: Double\n    open val m14: Double\n    open val
m21: Double\n    open val m22: Double\n    open val m23: Double\n    open val m24: Double\n    open val m31:
Double\n    open val m32: Double\n    open val m33: Double\n    open val m34: Double\n    open val m41: Double\n
    open val m42: Double\n    open val m43: Double\n    open val m44: Double\n    open val is2D: Boolean\n    open
val isIdentity: Boolean\n    fun translate(tx: Double, ty: Double, tz: Double = definedExternally): DOMMatrix\n
    fun scale(scale: Double, originX: Double = definedExternally, originY: Double = definedExternally): DOMMatrix\n
    fun scale3d(scale: Double, originX: Double = definedExternally, originY: Double = definedExternally, originZ:
Double = definedExternally): DOMMatrix\n    fun scaleNonUniform(scaleX: Double, scaleY: Double =
definedExternally, scaleZ: Double = definedExternally, originX: Double = definedExternally, originY: Double =

```

```

definedExternally, originZ: Double = definedExternally): DOMMatrix\n fun rotate(angle: Double, originX:
Double = definedExternally, originY: Double = definedExternally): DOMMatrix\n fun rotateFromVector(x:
Double, y: Double): DOMMatrix\n fun rotateAxisAngle(x: Double, y: Double, z: Double, angle: Double):
DOMMatrix\n fun skewX(sx: Double): DOMMatrix\n fun skewY(sy: Double): DOMMatrix\n fun
multiply(other: DOMMatrix): DOMMatrix\n fun flipX(): DOMMatrix\n fun flipY(): DOMMatrix\n fun
inverse(): DOMMatrix\n fun transformPoint(point: DOMPointInit = definedExternally): DOMPoint\n fun
toFloat32Array(): Float32Array\n fun toFloat64Array(): Float64Array\n}\n\n**\n * Exposes the JavaScript
[DOMMatrix](https://developer.mozilla.org/en/docs/Web/API/DOMMatrix) to Kotlin\n */\npublic external open
class DOMMatrix() : DOMMatrixReadOnly {\n constructor(transformList: String)\n constructor(other:
DOMMatrixReadOnly)\n constructor(array32: Float32Array)\n constructor(array64: Float64Array)\n
constructor(numberSequence: Array<Double>)\n override var a: Double\n override var b: Double\n override
var c: Double\n override var d: Double\n override var e: Double\n override var f: Double\n override var m11:
Double\n override var m12: Double\n override var m13: Double\n override var m14: Double\n override var
m21: Double\n override var m22: Double\n override var m23: Double\n override var m24: Double\n override
var m31: Double\n override var m32: Double\n override var m33: Double\n override var m34: Double\n
override var m41: Double\n override var m42: Double\n override var m43: Double\n override var m44:
Double\n fun multiplySelf(other: DOMMatrix): DOMMatrix\n fun preMultiplySelf(other: DOMMatrix):
DOMMatrix\n fun translateSelf(tx: Double, ty: Double, tz: Double = definedExternally): DOMMatrix\n fun
scaleSelf(scale: Double, originX: Double = definedExternally, originY: Double = definedExternally): DOMMatrix\n
fun scale3dSelf(scale: Double, originX: Double = definedExternally, originY: Double = definedExternally,
originZ: Double = definedExternally): DOMMatrix\n fun scaleNonUniformSelf(scaleX: Double, scaleY: Double =
definedExternally, scaleZ: Double = definedExternally, originX: Double = definedExternally, originY: Double =
definedExternally, originZ: Double = definedExternally): DOMMatrix\n fun rotateSelf(angle: Double, originX:
Double = definedExternally, originY: Double = definedExternally): DOMMatrix\n fun rotateFromVectorSelf(x:
Double, y: Double): DOMMatrix\n fun rotateAxisAngleSelf(x: Double, y: Double, z: Double, angle: Double):
DOMMatrix\n fun skewXSelf(sx: Double): DOMMatrix\n fun skewYSelf(sy: Double): DOMMatrix\n fun
invertSelf(): DOMMatrix\n fun setMatrixValue(transformList: String): DOMMatrix\n}\n\npublic external
interface ScrollOptions {\n var behavior: ScrollBehavior? /* = ScrollBehavior.AUTO */\n get() =
definedExternally\n set(value) = definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun ScrollOptions(behavior:
ScrollBehavior? = ScrollBehavior.AUTO): ScrollOptions {\n val o = js("{}")\n o["behavior"] = behavior\n
return o}\n}\n\n**\n * Exposes the JavaScript
[ScrollToOptions](https://developer.mozilla.org/en/docs/Web/API/ScrollToOptions) to Kotlin\n */\npublic external
interface ScrollToOptions : ScrollOptions {\n var left: Double?\n get() = definedExternally\n set(value) =
definedExternally\n var top: Double?\n get() = definedExternally\n set(value) =
definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun ScrollToOptions(left: Double? =
undefined, top: Double? = undefined, behavior: ScrollBehavior? = ScrollBehavior.AUTO): ScrollToOptions {\n
val o = js("{}")\n o["left"] = left\n o["top"] = top\n o["behavior"] = behavior\n return o}\n}\n\n**\n *
Exposes the JavaScript [MediaQueryList](https://developer.mozilla.org/en/docs/Web/API/MediaQueryList) to
Kotlin\n */\npublic external abstract class MediaQueryList : EventTarget {\n open val media: String\n open val
matches: Boolean\n open var onchange: ((Event) -> dynamic)?\n fun addListener(listener: EventListener?)\n
fun addListener(listener: ((Event) -> Unit)?)\n fun removeListener(listener: EventListener?)\n fun
removeListener(listener: ((Event) -> Unit)?)\n}\n\n**\n * Exposes the JavaScript
[MediaQueryListEvent](https://developer.mozilla.org/en/docs/Web/API/MediaQueryListEvent) to Kotlin\n
*/\npublic external open class MediaQueryListEvent(type: String, eventInitDict: MediaQueryListEventInit =
definedExternally) : Event {\n open val media: String\n open val matches: Boolean\n\n companion object {\n
val NONE: Short\n val CAPTURING_PHASE: Short\n val AT_TARGET: Short\n val

```

```

BUBBLING_PHASE: Short\n  }\n}\n\npublic external interface MediaQueryListEventInit : EventInit {\n  var
media: String? /* = \"\" */\n  get() = definedExternally\n  set(value) = definedExternally\n  var matches:
Boolean? /* = false */\n  get() = definedExternally\n  set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun MediaQueryListEventInit(media:
String? = \"\", matches: Boolean? = false, bubbles: Boolean? = false, cancelable: Boolean? = false, composed:
Boolean? = false): MediaQueryListEventInit {\n  val o = js(\"({})\")\n  o[\"media\"] = media\n  o[\"matches\"] =
matches\n  o[\"bubbles\"] = bubbles\n  o[\"cancelable\"] = cancelable\n  o[\"composed\"] = composed\n  return
o\n}\n\n/**\n * Exposes the JavaScript [Screen](https://developer.mozilla.org/en/docs/Web/API/Screen) to Kotlin\n
*/\n\npublic external abstract class Screen {\n  open val availWidth: Int\n  open val availHeight: Int\n  open val
width: Int\n  open val height: Int\n  open val colorDepth: Int\n  open val pixelDepth: Int\n}\n\n/**\n * Exposes
the JavaScript [CaretPosition](https://developer.mozilla.org/en/docs/Web/API/CaretPosition) to Kotlin\n
*/\n\npublic external abstract class CaretPosition {\n  open val offsetNode: Node\n  open val offset: Int\n  fun
getClientRect(): DOMRect?\n}\n\npublic external interface ScrollIntoViewOptions : ScrollOptions {\n  var block:
ScrollLogicalPosition? /* = ScrollLogicalPosition.CENTER */\n  get() = definedExternally\n  set(value) =
definedExternally\n  var inline: ScrollLogicalPosition? /* = ScrollLogicalPosition.CENTER */\n  get() =
definedExternally\n  set(value) = definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun ScrollIntoViewOptions(block:
ScrollLogicalPosition? = ScrollLogicalPosition.CENTER, inline: ScrollLogicalPosition? =
ScrollLogicalPosition.CENTER, behavior: ScrollBehavior? = ScrollBehavior.AUTO): ScrollIntoViewOptions {\n
val o = js(\"({})\")\n  o[\"block\"] = block\n  o[\"inline\"] = inline\n  o[\"behavior\"] = behavior\n  return
o\n}\n\npublic external interface BoxQuadOptions {\n  var box: CSSBoxType? /* = CSSBoxType.BORDER */\n
get() = definedExternally\n  set(value) = definedExternally\n  var relativeTo: dynamic\n  get() =
definedExternally\n  set(value) = definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun BoxQuadOptions(box: CSSBoxType?
= CSSBoxType.BORDER, relativeTo: dynamic = undefined): BoxQuadOptions {\n  val o = js(\"({})\")\n
o[\"box\"] = box\n  o[\"relativeTo\"] = relativeTo\n  return o\n}\n\npublic external interface
ConvertCoordinateOptions {\n  var fromBox: CSSBoxType? /* = CSSBoxType.BORDER */\n  get() =
definedExternally\n  set(value) = definedExternally\n  var toBox: CSSBoxType? /* = CSSBoxType.BORDER
*/\n  get() = definedExternally\n  set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun ConvertCoordinateOptions(fromBox:
CSSBoxType? = CSSBoxType.BORDER, toBox: CSSBoxType? = CSSBoxType.BORDER):
ConvertCoordinateOptions {\n  val o = js(\"({})\")\n  o[\"fromBox\"] = fromBox\n  o[\"toBox\"] = toBox\n
return o\n}\n\n/**\n * Exposes the JavaScript
[GeometryUtils](https://developer.mozilla.org/en/docs/Web/API/GeometryUtils) to Kotlin\n
*/\n\npublic external interface GeometryUtils {\n  fun getBoxQuads(options: BoxQuadOptions = definedExternally):
Array<DOMQuad>\n  fun convertQuadFromNode(quad: dynamic, from: dynamic, options:
ConvertCoordinateOptions = definedExternally): DOMQuad\n  fun convertRectFromNode(rect:
DOMRectReadOnly, from: dynamic, options: ConvertCoordinateOptions = definedExternally): DOMQuad\n  fun
convertPointFromNode(point: DOMPointInit, from: dynamic, options: ConvertCoordinateOptions =
definedExternally): DOMPoint\n}\n\n/**\n * Exposes the JavaScript
[Touch](https://developer.mozilla.org/en/docs/Web/API/Touch) to Kotlin\n
*/\n\npublic external abstract class Touch
{\n  open val identifier: Int\n  open val target: EventTarget\n  open val screenX: Int\n  open val screenY: Int\n
open val clientX: Int\n  open val clientY: Int\n  open val pageX: Int\n  open val pageY: Int\n  open val region:
String?\n}\n\npublic external abstract class TouchList : ItemArrayLike<Touch> {\n  override fun item(index: Int):
Touch?\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun TouchList.get(index: Int):

```

```

Touch? = asDynamic()[index]\n\npublic external open class TouchEvent : UIEvent {\n  open val touches:
TouchList\n  open val targetTouches: TouchList\n  open val changedTouches: TouchList\n  open val altKey:
Boolean\n  open val metaKey: Boolean\n  open val ctrlKey: Boolean\n  open val shiftKey: Boolean\n\n  companion object {\n    val NONE: Short\n    val CAPTURING_PHASE: Short\n    val AT_TARGET:
Short\n    val BUBBLING_PHASE: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[Image](https://developer.mozilla.org/en/docs/Web/API/Image) to Kotlin\n */\npublic external open class
Image(width: Int = definedExternally, height: Int = definedExternally) : HTMLElement {\n  override var
onabort: ((Event) -> dynamic)?\n  override var onblur: ((FocusEvent) -> dynamic)?\n  override var oncancel:
((Event) -> dynamic)?\n  override var oncanplay: ((Event) -> dynamic)?\n  override var oncanplaythrough:
((Event) -> dynamic)?\n  override var onchange: ((Event) -> dynamic)?\n  override var onclick: ((MouseEvent) ->
dynamic)?\n  override var onclose: ((Event) -> dynamic)?\n  override var oncontextmenu: ((MouseEvent) ->
dynamic)?\n  override var oncuechange: ((Event) -> dynamic)?\n  override var ondblclick: ((MouseEvent) ->
dynamic)?\n  override var ondrag: ((DragEvent) -> dynamic)?\n  override var ondragend: ((DragEvent) ->
dynamic)?\n  override var ondragenter: ((DragEvent) -> dynamic)?\n  override var ondragexit: ((DragEvent) ->
dynamic)?\n  override var ondragleave: ((DragEvent) -> dynamic)?\n  override var ondragover: ((DragEvent) ->
dynamic)?\n  override var ondragstart: ((DragEvent) -> dynamic)?\n  override var ondrop: ((DragEvent) ->
dynamic)?\n  override var ondurationchange: ((Event) -> dynamic)?\n  override var onemptied: ((Event) ->
dynamic)?\n  override var onended: ((Event) -> dynamic)?\n  override var onerror: ((dynamic, String, Int, Int,
Any?) -> dynamic)?\n  override var onfocus: ((FocusEvent) -> dynamic)?\n  override var oninput: ((InputEvent) ->
dynamic)?\n  override var oninvalid: ((Event) -> dynamic)?\n  override var onkeydown: ((KeyboardEvent) ->
dynamic)?\n  override var onkeypress: ((KeyboardEvent) -> dynamic)?\n  override var onkeyup:
((KeyboardEvent) -> dynamic)?\n  override var onload: ((Event) -> dynamic)?\n  override var onloadeddata:
((Event) -> dynamic)?\n  override var onloadedmetadata: ((Event) -> dynamic)?\n  override var onloadend:
((Event) -> dynamic)?\n  override var onloadstart: ((ProgressEvent) -> dynamic)?\n  override var onmousedown:
((MouseEvent) -> dynamic)?\n  override var onmouseenter: ((MouseEvent) -> dynamic)?\n  override var
onmouseleave: ((MouseEvent) -> dynamic)?\n  override var onmousemove: ((MouseEvent) -> dynamic)?\n
override var onmouseout: ((MouseEvent) -> dynamic)?\n  override var onmouseover: ((MouseEvent) ->
dynamic)?\n  override var onmouseup: ((MouseEvent) -> dynamic)?\n  override var onwheel: ((WheelEvent) ->
dynamic)?\n  override var onpause: ((Event) -> dynamic)?\n  override var onplay: ((Event) -> dynamic)?\n
override var onplaying: ((Event) -> dynamic)?\n  override var onprogress: ((ProgressEvent) -> dynamic)?\n
override var onratechange: ((Event) -> dynamic)?\n  override var onreset: ((Event) -> dynamic)?\n  override var
onresize: ((Event) -> dynamic)?\n  override var onscroll: ((Event) -> dynamic)?\n  override var onseeked:
((Event) -> dynamic)?\n  override var onseeking: ((Event) -> dynamic)?\n  override var onselect: ((Event) ->
dynamic)?\n  override var onshow: ((Event) -> dynamic)?\n  override var onstalled: ((Event) -> dynamic)?\n
override var onsubmit: ((Event) -> dynamic)?\n  override var onsuspend: ((Event) -> dynamic)?\n  override var
ontimeupdate: ((Event) -> dynamic)?\n  override var ontoggle: ((Event) -> dynamic)?\n  override var
onvolumechange: ((Event) -> dynamic)?\n  override var onwaiting: ((Event) -> dynamic)?\n  override var
ongotpointercapture: ((PointerEvent) -> dynamic)?\n  override var onlostpointercapture: ((PointerEvent) ->
dynamic)?\n  override var onpointerdown: ((PointerEvent) -> dynamic)?\n  override var onpointermove:
((PointerEvent) -> dynamic)?\n  override var onpointerup: ((PointerEvent) -> dynamic)?\n  override var
onpointercancel: ((PointerEvent) -> dynamic)?\n  override var onpointerover: ((PointerEvent) -> dynamic)?\n
override var onpointerout: ((PointerEvent) -> dynamic)?\n  override var onpointerenter: ((PointerEvent) ->
dynamic)?\n  override var onpointerleave: ((PointerEvent) -> dynamic)?\n  override var oncopy:
((ClipboardEvent) -> dynamic)?\n  override var oncut: ((ClipboardEvent) -> dynamic)?\n  override var onpaste:
((ClipboardEvent) -> dynamic)?\n  override var contentEditable: String\n  override val isContentEditable:
Boolean\n  override val style: CSSStyleDeclaration\n  override val children: HTMLCollection\n  override val
firstElementChild: Element?\n  override val lastElementChild: Element?\n  override val childElementCount: Int\n
  override val previousElementSibling: Element?\n  override val nextElementSibling: Element?\n  override val

```

```

assignedSlot: HTMLSlotElement? \n  override fun prepend(vararg nodes: dynamic) \n  override fun append(vararg
nodes: dynamic) \n  override fun querySelector(selectors: String): Element? \n  override fun
querySelectorAll(selectors: String): NodeList \n  override fun before(vararg nodes: dynamic) \n  override fun
after(vararg nodes: dynamic) \n  override fun replaceWith(vararg nodes: dynamic) \n  override fun remove() \n
override fun getBoxQuads(options: BoxQuadOptions /* = definedExternally */): Array<DOMQuad> \n  override
fun convertQuadFromNode(quad: dynamic, from: dynamic, options: ConvertCoordinateOptions /* =
definedExternally */): DOMQuad \n  override fun convertRectFromNode(rect: DOMRectReadOnly, from:
dynamic, options: ConvertCoordinateOptions /* = definedExternally */): DOMQuad \n  override fun
convertPointFromNode(point: DOMPointInit, from: dynamic, options: ConvertCoordinateOptions /* =
definedExternally */): DOMPoint \n \n  companion object { \n      val ELEMENT_NODE: Short \n      val
ATTRIBUTE_NODE: Short \n      val TEXT_NODE: Short \n      val CDATA_SECTION_NODE: Short \n      val
ENTITY_REFERENCE_NODE: Short \n      val ENTITY_NODE: Short \n      val
PROCESSING_INSTRUCTION_NODE: Short \n      val COMMENT_NODE: Short \n      val
DOCUMENT_NODE: Short \n      val DOCUMENT_TYPE_NODE: Short \n      val
DOCUMENT_FRAGMENT_NODE: Short \n      val NOTATION_NODE: Short \n      val
DOCUMENT_POSITION_DISCONNECTED: Short \n      val DOCUMENT_POSITION_PRECEDING: Short \n
      val DOCUMENT_POSITION_FOLLOWING: Short \n      val DOCUMENT_POSITION_CONTAINS: Short \n
      val DOCUMENT_POSITION_CONTAINED_BY: Short \n      val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short \n      } \n \n public external open class
Audio(src: String = definedExternally) : HTMLAudioElement { \n  override var onabort: ((Event) -> dynamic)? \n
override var onblur: ((FocusEvent) -> dynamic)? \n  override var oncancel: ((Event) -> dynamic)? \n  override var
oncanplay: ((Event) -> dynamic)? \n  override var oncanplaythrough: ((Event) -> dynamic)? \n  override var
onchange: ((Event) -> dynamic)? \n  override var onclick: ((MouseEvent) -> dynamic)? \n  override var onclose:
((Event) -> dynamic)? \n  override var oncontextmenu: ((MouseEvent) -> dynamic)? \n  override var oncuechange:
((Event) -> dynamic)? \n  override var ondblclick: ((MouseEvent) -> dynamic)? \n  override var ondrag:
((DragEvent) -> dynamic)? \n  override var ondragend: ((DragEvent) -> dynamic)? \n  override var ondragenter:
((DragEvent) -> dynamic)? \n  override var ondragexit: ((DragEvent) -> dynamic)? \n  override var ondragleave:
((DragEvent) -> dynamic)? \n  override var ondragover: ((DragEvent) -> dynamic)? \n  override var ondragstart:
((DragEvent) -> dynamic)? \n  override var ondrop: ((DragEvent) -> dynamic)? \n  override var ondurationchange:
((Event) -> dynamic)? \n  override var onemptied: ((Event) -> dynamic)? \n  override var onended: ((Event) ->
dynamic)? \n  override var onerror: ((dynamic, String, Int, Int, Any?) -> dynamic)? \n  override var onfocus:
((FocusEvent) -> dynamic)? \n  override var oninput: ((InputEvent) -> dynamic)? \n  override var oninvalid:
((Event) -> dynamic)? \n  override var onkeydown: ((KeyboardEvent) -> dynamic)? \n  override var onkeypress:
((KeyboardEvent) -> dynamic)? \n  override var onkeyup: ((KeyboardEvent) -> dynamic)? \n  override var onload:
((Event) -> dynamic)? \n  override var onloadeddata: ((Event) -> dynamic)? \n  override var onloadedmetadata:
((Event) -> dynamic)? \n  override var onloadend: ((Event) -> dynamic)? \n  override var onloadstart:
((ProgressEvent) -> dynamic)? \n  override var onmousedown: ((MouseEvent) -> dynamic)? \n  override var
onmouseenter: ((MouseEvent) -> dynamic)? \n  override var onmouseleave: ((MouseEvent) -> dynamic)? \n
override var onmousemove: ((MouseEvent) -> dynamic)? \n  override var onmouseout: ((MouseEvent) ->
dynamic)? \n  override var onmouseover: ((MouseEvent) -> dynamic)? \n  override var onmouseup: ((MouseEvent)
-> dynamic)? \n  override var onwheel: ((WheelEvent) -> dynamic)? \n  override var onpause: ((Event) ->
dynamic)? \n  override var onplay: ((Event) -> dynamic)? \n  override var onplaying: ((Event) -> dynamic)? \n
override var onprogress: ((ProgressEvent) -> dynamic)? \n  override var onratechange: ((Event) -> dynamic)? \n
override var onreset: ((Event) -> dynamic)? \n  override var onresize: ((Event) -> dynamic)? \n  override var
onscroll: ((Event) -> dynamic)? \n  override var onseeked: ((Event) -> dynamic)? \n  override var onseeking:
((Event) -> dynamic)? \n  override var onselect: ((Event) -> dynamic)? \n  override var onshow: ((Event) ->
dynamic)? \n  override var onstalled: ((Event) -> dynamic)? \n  override var onsubmit: ((Event) -> dynamic)? \n
override var onsuspend: ((Event) -> dynamic)? \n  override var ontimeupdate: ((Event) -> dynamic)? \n  override

```

```

var ontoggle: ((Event) -> dynamic)?\n  override var onvolumechange: ((Event) -> dynamic)?\n  override var
onwaiting: ((Event) -> dynamic)?\n  override var ongotpointercapture: ((PointerEvent) -> dynamic)?\n  override
var onlostpointercapture: ((PointerEvent) -> dynamic)?\n  override var onpointerdown: ((PointerEvent) ->
dynamic)?\n  override var onpointermove: ((PointerEvent) -> dynamic)?\n  override var onpointerup:
((PointerEvent) -> dynamic)?\n  override var onpointercancel: ((PointerEvent) -> dynamic)?\n  override var
onpointerover: ((PointerEvent) -> dynamic)?\n  override var onpointerout: ((PointerEvent) -> dynamic)?\n
override var onpointerenter: ((PointerEvent) -> dynamic)?\n  override var onpointerleave: ((PointerEvent) ->
dynamic)?\n  override var oncopy: ((ClipboardEvent) -> dynamic)?\n  override var oncut: ((ClipboardEvent) ->
dynamic)?\n  override var onpaste: ((ClipboardEvent) -> dynamic)?\n  override var contentEditable: String\n
override val isContentEditable: Boolean\n  override val style: CSSStyleDeclaration\n  override val children:
HTMLCollection\n  override val firstElementChild: Element?\n  override val lastElementChild: Element?\n
override val childElementCount: Int\n  override val previousElementSibling: Element?\n  override val
nextElementSibling: Element?\n  override val assignedSlot: HTMLSlotElement?\n  override fun prepend(vararg
nodes: dynamic)\n  override fun append(vararg nodes: dynamic)\n  override fun querySelector(selectors: String):
Element?\n  override fun querySelectorAll(selectors: String): NodeList\n  override fun before(vararg nodes:
dynamic)\n  override fun after(vararg nodes: dynamic)\n  override fun replaceWith(vararg nodes: dynamic)\n
override fun remove()\n  override fun getBoxQuads(options: BoxQuadOptions /* = definedExternally */):
Array<DOMQuad>\n  override fun convertQuadFromNode(quad: dynamic, from: dynamic, options:
ConvertCoordinateOptions /* = definedExternally */): DOMQuad\n  override fun convertRectFromNode(rect:
DOMRectReadOnly, from: dynamic, options: ConvertCoordinateOptions /* = definedExternally */): DOMQuad\n
override fun convertPointFromNode(point: DOMPointInit, from: dynamic, options: ConvertCoordinateOptions /* =
definedExternally */): DOMPoint\n\n  companion object {\n    val NETWORK_EMPTY: Short\n    val
NETWORK_IDLE: Short\n    val NETWORK_LOADING: Short\n    val NETWORK_NO_SOURCE: Short\n
    val HAVE_NOTHING: Short\n    val HAVE_METADATA: Short\n    val HAVE_CURRENT_DATA:
Short\n    val HAVE_FUTURE_DATA: Short\n    val HAVE_ENOUGH_DATA: Short\n    val
ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val
CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE:
Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[Option](https://developer.mozilla.org/en/docs/Web/API/Option) to Kotlin\n */\n\npublic external open class
Option(text: String = definedExternally, value: String = definedExternally, defaultSelected: Boolean =
definedExternally, selected: Boolean = definedExternally) : HTMLOptionElement {\n  override var onabort:
((Event) -> dynamic)?\n  override var onblur: ((FocusEvent) -> dynamic)?\n  override var oncancel: ((Event) ->
dynamic)?\n  override var oncanplay: ((Event) -> dynamic)?\n  override var oncanplaythrough: ((Event) ->
dynamic)?\n  override var onchange: ((Event) -> dynamic)?\n  override var onclick: ((MouseEvent) ->
dynamic)?\n  override var onclose: ((Event) -> dynamic)?\n  override var oncontextmenu: ((MouseEvent) ->
dynamic)?\n  override var oncuechange: ((Event) -> dynamic)?\n  override var ondblclick: ((MouseEvent) ->
dynamic)?\n  override var ondrag: ((DragEvent) -> dynamic)?\n  override var ondragend: ((DragEvent) ->
dynamic)?\n  override var ondragenter: ((DragEvent) -> dynamic)?\n  override var ondragexit: ((DragEvent) ->
dynamic)?\n  override var ondragleave: ((DragEvent) -> dynamic)?\n  override var ondragover: ((DragEvent) ->
dynamic)?\n  override var ondragstart: ((DragEvent) -> dynamic)?\n  override var ondrop: ((DragEvent) ->
dynamic)?\n  override var ondurationchange: ((Event) -> dynamic)?\n  override var onemptied: ((Event) ->
dynamic)?\n  override var onended: ((Event) -> dynamic)?\n  override var onerror: ((dynamic, String, Int, Int,

```

```

Any?) -> dynamic)?\n  override var onfocus: ((FocusEvent) -> dynamic)?\n  override var oninput: ((InputEvent) ->
dynamic)?\n  override var oninvalid: ((Event) -> dynamic)?\n  override var onkeydown: ((KeyboardEvent) ->
dynamic)?\n  override var onkeypress: ((KeyboardEvent) -> dynamic)?\n  override var onkeyup:
((KeyboardEvent) -> dynamic)?\n  override var onload: ((Event) -> dynamic)?\n  override var onloadeddata:
((Event) -> dynamic)?\n  override var onloadedmetadata: ((Event) -> dynamic)?\n  override var onloadend:
((Event) -> dynamic)?\n  override var onloadstart: ((ProgressEvent) -> dynamic)?\n  override var onmousedown:
((MouseEvent) -> dynamic)?\n  override var onmouseenter: ((MouseEvent) -> dynamic)?\n  override var
onmouseleave: ((MouseEvent) -> dynamic)?\n  override var onmousemove: ((MouseEvent) -> dynamic)?\n
override var onmouseout: ((MouseEvent) -> dynamic)?\n  override var onmouseover: ((MouseEvent) ->
dynamic)?\n  override var onmouseup: ((MouseEvent) -> dynamic)?\n  override var onwheel: ((WheelEvent) ->
dynamic)?\n  override var onpause: ((Event) -> dynamic)?\n  override var onplay: ((Event) -> dynamic)?\n
override var onplaying: ((Event) -> dynamic)?\n  override var onprogress: ((ProgressEvent) -> dynamic)?\n
override var onratechange: ((Event) -> dynamic)?\n  override var onreset: ((Event) -> dynamic)?\n  override var
onresize: ((Event) -> dynamic)?\n  override var onscroll: ((Event) -> dynamic)?\n  override var onseeked:
((Event) -> dynamic)?\n  override var onseeking: ((Event) -> dynamic)?\n  override var onselect: ((Event) ->
dynamic)?\n  override var onshow: ((Event) -> dynamic)?\n  override var onstalled: ((Event) -> dynamic)?\n
override var onsubmit: ((Event) -> dynamic)?\n  override var onsuspend: ((Event) -> dynamic)?\n  override var
ontimeupdate: ((Event) -> dynamic)?\n  override var ontoggle: ((Event) -> dynamic)?\n  override var
onvolumechange: ((Event) -> dynamic)?\n  override var onwaiting: ((Event) -> dynamic)?\n  override var
ongotpointercapture: ((PointerEvent) -> dynamic)?\n  override var onlostpointercapture: ((PointerEvent) ->
dynamic)?\n  override var onpointerdown: ((PointerEvent) -> dynamic)?\n  override var onpointermove:
((PointerEvent) -> dynamic)?\n  override var onpointerup: ((PointerEvent) -> dynamic)?\n  override var
onpointercancel: ((PointerEvent) -> dynamic)?\n  override var onpointerover: ((PointerEvent) -> dynamic)?\n
override var onpointerout: ((PointerEvent) -> dynamic)?\n  override var onpointerenter: ((PointerEvent) ->
dynamic)?\n  override var onpointerleave: ((PointerEvent) -> dynamic)?\n  override var oncopy:
((ClipboardEvent) -> dynamic)?\n  override var oncut: ((ClipboardEvent) -> dynamic)?\n  override var onpaste:
((ClipboardEvent) -> dynamic)?\n  override var contentEditable: String\n  override val isContentEditable:
Boolean\n  override val style: CSSStyleDeclaration\n  override val children: HTMLCollection\n  override val
firstElementChild: Element?\n  override val lastElementChild: Element?\n  override val childElementCount: Int\n
  override val previousElementSibling: Element?\n  override val nextElementSibling: Element?\n  override val
assignedSlot: HTMLSlotElement?\n  override fun prepend(vararg nodes: dynamic)\n  override fun append(vararg
nodes: dynamic)\n  override fun querySelector(selectors: String): Element?\n  override fun
querySelectorAll(selectors: String): NodeList\n  override fun before(vararg nodes: dynamic)\n  override fun
after(vararg nodes: dynamic)\n  override fun replaceWith(vararg nodes: dynamic)\n  override fun remove()\n
  override fun getBoxQuads(options: BoxQuadOptions /* = definedExternally */): Array<DOMQuad>\n  override
fun convertQuadFromNode(quad: dynamic, from: dynamic, options: ConvertCoordinateOptions /* =
definedExternally */): DOMQuad\n  override fun convertRectFromNode(rect: DOMRectReadOnly, from:
dynamic, options: ConvertCoordinateOptions /* = definedExternally */): DOMQuad\n  override fun
convertPointFromNode(point: DOMPointInit, from: dynamic, options: ConvertCoordinateOptions /* =
definedExternally */): DOMPoint\n\n  companion object {\n    val ELEMENT_NODE: Short\n    val
ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val

```

```

DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\npublic external interface
UnionElementOrHTMLCollection\n\npublic external interface UnionElementOrRadioNodeList\n\npublic external
interface UnionHTMLOptGroupElementOrHTMLOptionElement\n\npublic external interface
UnionAudioTrackOrTextTrackOrVideoTrack\n\npublic external interface UnionElementOrMouseEvent\n\npublic
external interface UnionMessagePortOrWindowProxy\n\npublic external interface MediaPlayer\n\npublic
external interface RenderingContext\n\npublic external interface HTMLOrSVGImageElement :
CanvasImageSource\n\npublic external interface CanvasImageSource : ImageBitmapSource\n\npublic external
interface ImageBitmapSource\n\npublic external interface HTMLOrSVGScriptElement\n\n/* please, don't
implement this interface!
*\n@JsName("\null")\n@Suppress("\NESTED_CLASS_IN_EXTERNAL_INTERFACE")\n\npublic external
interface DocumentReadyState {\n  companion object\n}\n\npublic inline val
DocumentReadyState.Companion.LOADING: DocumentReadyState get() =
"loading".asDynamic().unsafeCast<DocumentReadyState>()\n\npublic inline val
DocumentReadyState.Companion.INTERACTIVE: DocumentReadyState get() =
"interactive".asDynamic().unsafeCast<DocumentReadyState>()\n\npublic inline val
DocumentReadyState.Companion.COMPLETE: DocumentReadyState get() =
"complete".asDynamic().unsafeCast<DocumentReadyState>()\n\n/* please, don't implement this interface!
*\n@JsName("\null")\n@Suppress("\NESTED_CLASS_IN_EXTERNAL_INTERFACE")\n\npublic external
interface CanPlayTypeResult {\n  companion object\n}\n\npublic inline val
CanPlayTypeResult.Companion.EMPTY: CanPlayTypeResult get() =
"".asDynamic().unsafeCast<CanPlayTypeResult>()\n\npublic inline val CanPlayTypeResult.Companion.MAYBE:
CanPlayTypeResult get() = "maybe".asDynamic().unsafeCast<CanPlayTypeResult>()\n\npublic inline val
CanPlayTypeResult.Companion.PROBABLY: CanPlayTypeResult get() =
"probably".asDynamic().unsafeCast<CanPlayTypeResult>()\n\n/* please, don't implement this interface!
*\n@JsName("\null")\n@Suppress("\NESTED_CLASS_IN_EXTERNAL_INTERFACE")\n\npublic external
interface TextTrackMode {\n  companion object\n}\n\npublic inline val TextTrackMode.Companion.DISABLED:
TextTrackMode get() = "disabled".asDynamic().unsafeCast<TextTrackMode>()\n\npublic inline val
TextTrackMode.Companion.HIDDEN: TextTrackMode get() =
"hidden".asDynamic().unsafeCast<TextTrackMode>()\n\npublic inline val
TextTrackMode.Companion.SHOWING: TextTrackMode get() =
"showing".asDynamic().unsafeCast<TextTrackMode>()\n\n/* please, don't implement this interface!
*\n@JsName("\null")\n@Suppress("\NESTED_CLASS_IN_EXTERNAL_INTERFACE")\n\npublic external
interface TextTrackKind {\n  companion object\n}\n\npublic inline val TextTrackKind.Companion.SUBTITLES:
TextTrackKind get() = "subtitles".asDynamic().unsafeCast<TextTrackKind>()\n\npublic inline val
TextTrackKind.Companion.CAPTIONS: TextTrackKind get() =
"captions".asDynamic().unsafeCast<TextTrackKind>()\n\npublic inline val
TextTrackKind.Companion.DESCRPTIONS: TextTrackKind get() =
"descriptions".asDynamic().unsafeCast<TextTrackKind>()\n\npublic inline val
TextTrackKind.Companion.CHAPTERS: TextTrackKind get() =
"chapters".asDynamic().unsafeCast<TextTrackKind>()\n\npublic inline val
TextTrackKind.Companion.METADATA: TextTrackKind get() =
"metadata".asDynamic().unsafeCast<TextTrackKind>()\n\n/* please, don't implement this interface!
*\n@JsName("\null")\n@Suppress("\NESTED_CLASS_IN_EXTERNAL_INTERFACE")\n\npublic external
interface SelectionMode {\n  companion object\n}\n\npublic inline val SelectionMode.Companion.SELECT:
SelectionMode get() = "select".asDynamic().unsafeCast<SelectionMode>()\n\npublic inline val
SelectionMode.Companion.START: SelectionMode get() =
"start".asDynamic().unsafeCast<SelectionMode>()\n\npublic inline val SelectionMode.Companion.END:
SelectionMode get() = "end".asDynamic().unsafeCast<SelectionMode>()\n\npublic inline val

```

```

SelectionMode.Companion.PRESERVE: SelectionMode get() =
`preserve`.asDynamic().unsafeCast<SelectionMode>()\n\n/* please, don't implement this interface!
*\n\n@JsName(`null`)\n\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic external
interface CanvasFillRule {\n  companion object\n}\n\npublic inline val CanvasFillRule.Companion.NONZERO:
CanvasFillRule get() = `nonzero`.asDynamic().unsafeCast<CanvasFillRule>()\n\npublic inline val
CanvasFillRule.Companion.EVENODD: CanvasFillRule get() =
`evenodd`.asDynamic().unsafeCast<CanvasFillRule>()\n\n/* please, don't implement this interface!
*\n\n@JsName(`null`)\n\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic external
interface ImageSmoothingQuality {\n  companion object\n}\n\npublic inline val
ImageSmoothingQuality.Companion.LOW: ImageSmoothingQuality get() =
`low`.asDynamic().unsafeCast<ImageSmoothingQuality>()\n\npublic inline val
ImageSmoothingQuality.Companion.MEDIUM: ImageSmoothingQuality get() =
`medium`.asDynamic().unsafeCast<ImageSmoothingQuality>()\n\npublic inline val
ImageSmoothingQuality.Companion.HIGH: ImageSmoothingQuality get() =
`high`.asDynamic().unsafeCast<ImageSmoothingQuality>()\n\n/* please, don't implement this interface!
*\n\n@JsName(`null`)\n\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic external
interface CanvasLineCap {\n  companion object\n}\n\npublic inline val CanvasLineCap.Companion.BUTT:
CanvasLineCap get() = `butt`.asDynamic().unsafeCast<CanvasLineCap>()\n\npublic inline val
CanvasLineCap.Companion.ROUND: CanvasLineCap get() =
`round`.asDynamic().unsafeCast<CanvasLineCap>()\n\npublic inline val CanvasLineCap.Companion.SQUARE:
CanvasLineCap get() = `square`.asDynamic().unsafeCast<CanvasLineCap>()\n\n/* please, don't implement this
interface! *\n\n@JsName(`null`)\n\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic
external interface CanvasLineJoin {\n  companion object\n}\n\npublic inline val
CanvasLineJoin.Companion.ROUND: CanvasLineJoin get() =
`round`.asDynamic().unsafeCast<CanvasLineJoin>()\n\npublic inline val CanvasLineJoin.Companion.BEVEL:
CanvasLineJoin get() = `bevel`.asDynamic().unsafeCast<CanvasLineJoin>()\n\npublic inline val
CanvasLineJoin.Companion.MITER: CanvasLineJoin get() =
`miter`.asDynamic().unsafeCast<CanvasLineJoin>()\n\n/* please, don't implement this interface!
*\n\n@JsName(`null`)\n\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic external
interface CanvasTextAlign {\n  companion object\n}\n\npublic inline val CanvasTextAlign.Companion.START:
CanvasTextAlign get() = `start`.asDynamic().unsafeCast<CanvasTextAlign>()\n\npublic inline val
CanvasTextAlign.Companion.END: CanvasTextAlign get() =
`end`.asDynamic().unsafeCast<CanvasTextAlign>()\n\npublic inline val CanvasTextAlign.Companion.LEFT:
CanvasTextAlign get() = `left`.asDynamic().unsafeCast<CanvasTextAlign>()\n\npublic inline val
CanvasTextAlign.Companion.RIGHT: CanvasTextAlign get() =
`right`.asDynamic().unsafeCast<CanvasTextAlign>()\n\npublic inline val
CanvasTextAlign.Companion.CENTER: CanvasTextAlign get() =
`center`.asDynamic().unsafeCast<CanvasTextAlign>()\n\n/* please, don't implement this interface!
*\n\n@JsName(`null`)\n\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic external
interface CanvasTextBaseline {\n  companion object\n}\n\npublic inline val CanvasTextBaseline.Companion.TOP:
CanvasTextBaseline get() = `top`.asDynamic().unsafeCast<CanvasTextBaseline>()\n\npublic inline val
CanvasTextBaseline.Companion.HANGING: CanvasTextBaseline get() =
`hanging`.asDynamic().unsafeCast<CanvasTextBaseline>()\n\npublic inline val
CanvasTextBaseline.Companion.MIDDLE: CanvasTextBaseline get() =
`middle`.asDynamic().unsafeCast<CanvasTextBaseline>()\n\npublic inline val
CanvasTextBaseline.Companion.ALPHABETIC: CanvasTextBaseline get() =
`alphabetic`.asDynamic().unsafeCast<CanvasTextBaseline>()\n\npublic inline val
CanvasTextBaseline.Companion.IDEOGRAPHIC: CanvasTextBaseline get() =

```

```

\"ideographic\".asDynamic().unsafeCast<CanvasTextBaseline>()\n\npublic inline val
CanvasTextBaseline.Companion.BOTTOM: CanvasTextBaseline get() =
\"bottom\".asDynamic().unsafeCast<CanvasTextBaseline>()\n\n/* please, don't implement this interface!
*\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\n\npublic external
interface CanvasDirection {\n    companion object\n}\n\npublic inline val CanvasDirection.Companion.LTR:
CanvasDirection get() = \"ltr\".asDynamic().unsafeCast<CanvasDirection>()\n\npublic inline val
CanvasDirection.Companion.RTL: CanvasDirection get() =
\"rtl\".asDynamic().unsafeCast<CanvasDirection>()\n\npublic inline val CanvasDirection.Companion.INHERIT:
CanvasDirection get() = \"inherit\".asDynamic().unsafeCast<CanvasDirection>()\n\n/* please, don't implement this
interface! *\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\n\npublic
external interface ScrollRestoration {\n    companion object\n}\n\npublic inline val
ScrollRestoration.Companion.AUTO: ScrollRestoration get() =
\"auto\".asDynamic().unsafeCast<ScrollRestoration>()\n\npublic inline val
ScrollRestoration.Companion.MANUAL: ScrollRestoration get() =
\"manual\".asDynamic().unsafeCast<ScrollRestoration>()\n\n/* please, don't implement this interface!
*\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\n\npublic external
interface ImageOrientation {\n    companion object\n}\n\npublic inline val ImageOrientation.Companion.NONE:
ImageOrientation get() = \"none\".asDynamic().unsafeCast<ImageOrientation>()\n\npublic inline val
ImageOrientation.Companion.FLIPY: ImageOrientation get() =
\"flipY\".asDynamic().unsafeCast<ImageOrientation>()\n\n/* please, don't implement this interface!
*\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\n\npublic external
interface PremultiplyAlpha {\n    companion object\n}\n\npublic inline val PremultiplyAlpha.Companion.NONE:
PremultiplyAlpha get() = \"none\".asDynamic().unsafeCast<PremultiplyAlpha>()\n\npublic inline val
PremultiplyAlpha.Companion.PREMULTIPLY: PremultiplyAlpha get() =
\"premultiply\".asDynamic().unsafeCast<PremultiplyAlpha>()\n\npublic inline val
PremultiplyAlpha.Companion.DEFAULT: PremultiplyAlpha get() =
\"default\".asDynamic().unsafeCast<PremultiplyAlpha>()\n\n/* please, don't implement this interface!
*\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\n\npublic external
interface ColorSpaceConversion {\n    companion object\n}\n\npublic inline val
ColorSpaceConversion.Companion.NONE: ColorSpaceConversion get() =
\"none\".asDynamic().unsafeCast<ColorSpaceConversion>()\n\npublic inline val
ColorSpaceConversion.Companion.DEFAULT: ColorSpaceConversion get() =
\"default\".asDynamic().unsafeCast<ColorSpaceConversion>()\n\n/* please, don't implement this interface!
*\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\n\npublic external
interface ResizeQuality {\n    companion object\n}\n\npublic inline val ResizeQuality.Companion.PIXELATED:
ResizeQuality get() = \"pixelated\".asDynamic().unsafeCast<ResizeQuality>()\n\npublic inline val
ResizeQuality.Companion.LOW: ResizeQuality get() =
\"low\".asDynamic().unsafeCast<ResizeQuality>()\n\npublic inline val ResizeQuality.Companion.MEDIUM:
ResizeQuality get() = \"medium\".asDynamic().unsafeCast<ResizeQuality>()\n\npublic inline val
ResizeQuality.Companion.HIGH: ResizeQuality get() = \"high\".asDynamic().unsafeCast<ResizeQuality>()\n\n/*
please, don't implement this interface!
*\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\n\npublic external
interface BinaryType {\n    companion object\n}\n\npublic inline val BinaryType.Companion.BLOB: BinaryType
get() = \"blob\".asDynamic().unsafeCast<BinaryType>()\n\npublic inline val
BinaryType.Companion.ARRAYBUFFER: BinaryType get() =
\"arraybuffer\".asDynamic().unsafeCast<BinaryType>()\n\n/* please, don't implement this interface!
*\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\n\npublic external
interface WorkerType {\n    companion object\n}\n\npublic inline val WorkerType.Companion.CLASSIC:

```

```

WorkerType get() = `classic`.asDynamic().unsafeCast<WorkerType>()\n\npublic inline val
WorkerType.Companion.MODULE: WorkerType get() =
`module`.asDynamic().unsafeCast<WorkerType>()\n\n/* please, don't implement this interface!
*\n@JsName(`null`)\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic external
interface ShadowRootMode {\n  companion object\n}\n\npublic inline val ShadowRootMode.Companion.OPEN:
ShadowRootMode get() = `open`.asDynamic().unsafeCast<ShadowRootMode>()\n\npublic inline val
ShadowRootMode.Companion.CLOSED: ShadowRootMode get() =
`closed`.asDynamic().unsafeCast<ShadowRootMode>()\n\n/* please, don't implement this interface!
*\n@JsName(`null`)\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic external
interface ScrollBehavior {\n  companion object\n}\n\npublic inline val ScrollBehavior.Companion.AUTO:
ScrollBehavior get() = `auto`.asDynamic().unsafeCast<ScrollBehavior>()\n\npublic inline val
ScrollBehavior.Companion.INSTANT: ScrollBehavior get() =
`instant`.asDynamic().unsafeCast<ScrollBehavior>()\n\npublic inline val ScrollBehavior.Companion.SMOOTH:
ScrollBehavior get() = `smooth`.asDynamic().unsafeCast<ScrollBehavior>()\n\n/* please, don't implement this
interface! *\n@JsName(`null`)\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic
external interface ScrollLogicalPosition {\n  companion object\n}\n\npublic inline val
ScrollLogicalPosition.Companion.START: ScrollLogicalPosition get() =
`start`.asDynamic().unsafeCast<ScrollLogicalPosition>()\n\npublic inline val
ScrollLogicalPosition.Companion.CENTER: ScrollLogicalPosition get() =
`center`.asDynamic().unsafeCast<ScrollLogicalPosition>()\n\npublic inline val
ScrollLogicalPosition.Companion.END: ScrollLogicalPosition get() =
`end`.asDynamic().unsafeCast<ScrollLogicalPosition>()\n\npublic inline val
ScrollLogicalPosition.Companion.NEAREST: ScrollLogicalPosition get() =
`nearest`.asDynamic().unsafeCast<ScrollLogicalPosition>()\n\n/* please, don't implement this interface!
*\n@JsName(`null`)\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic external
interface CSSBoxType {\n  companion object\n}\n\npublic inline val CSSBoxType.Companion.MARGIN:
CSSBoxType get() = `margin`.asDynamic().unsafeCast<CSSBoxType>()\n\npublic inline val
CSSBoxType.Companion.BORDER: CSSBoxType get() =
`border`.asDynamic().unsafeCast<CSSBoxType>()\n\npublic inline val CSSBoxType.Companion.PADDING:
CSSBoxType get() = `padding`.asDynamic().unsafeCast<CSSBoxType>()\n\npublic inline val
CSSBoxType.Companion.CONTENT: CSSBoxType get() =
`content`.asDynamic().unsafeCast<CSSBoxType>()"/\n\n/* Copyright 2010-2021 JetBrains s.r.o. and Kotlin
Programming Language contributors.\n\n/* Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.\n\n*\n\n// NOTE: THIS FILE IS AUTO-GENERATED, DO NOT EDIT!\n\n//
See github.com/kotlin/dukat for details\n\npackage org.w3c.fetch\n\nimport kotlin.js.*\nimport
org.khronos.webgl.*\nimport org.w3c.files.*\nimport org.w3c.xhr.*\n\n/**\n\n * Exposes the JavaScript
[Headers](https://developer.mozilla.org/en/docs/Web/API/Headers) to Kotlin\n\n *\n\npublic external open class
Headers(init: dynamic = definedExternally) {\n  fun append(name: String, value: String)\n  fun delete(name:
String)\n  fun get(name: String): String?\n  fun has(name: String): Boolean\n  fun set(name: String, value:
String)\n}\n\n\n/**\n\n * Exposes the JavaScript [Body](https://developer.mozilla.org/en/docs/Web/API/Body) to
Kotlin\n\n *\n\npublic external interface Body {\n  val bodyUsed: Boolean\n  fun ArrayBuffer():
Promise<ArrayBuffer>\n  fun blob(): Promise<Blob>\n  fun formData(): Promise<FormData>\n  fun json():
Promise<Any?>\n  fun text(): Promise<String>\n}\n\n\n/**\n\n * Exposes the JavaScript
[Request](https://developer.mozilla.org/en/docs/Web/API/Request) to Kotlin\n\n *\n\npublic external open class
Request(input: dynamic, init: RequestInit = definedExternally) : Body {\n  open val method: String\n  open val
url: String\n  open val headers: Headers\n  open val type: RequestType\n  open val destination:
RequestDestination\n  open val referrer: String\n  open val referrerPolicy: dynamic\n  open val mode:
RequestMode\n  open val credentials: RequestCredentials\n  open val cache: RequestCache\n  open val redirect:

```

```

RequestRedirect\n  open val integrity: String\n  open val keepalive: Boolean\n  override val bodyUsed:
Boolean\n  fun clone(): Request\n  override fun arrayBuffer(): Promise<ArrayBuffer>\n  override fun blob():
Promise<Blob>\n  override fun formData(): Promise<FormData>\n  override fun json(): Promise<Any?>\n
override fun text(): Promise<String>\n}\n\npublic external interface RequestInit {\n  var method: String?\n
get() = definedExternally\n  set(value) = definedExternally\n  var headers: dynamic\n  get() =
definedExternally\n  set(value) = definedExternally\n  var body: dynamic\n  get() = definedExternally\n
set(value) = definedExternally\n  var referrer: String?\n  get() = definedExternally\n  set(value) =
definedExternally\n  var referrerPolicy: dynamic\n  get() = definedExternally\n  set(value) =
definedExternally\n  var mode: RequestMode?\n  get() = definedExternally\n  set(value) =
definedExternally\n  var credentials: RequestCredentials?\n  get() = definedExternally\n  set(value) =
definedExternally\n  var cache: RequestCache?\n  get() = definedExternally\n  set(value) =
definedExternally\n  var redirect: RequestRedirect?\n  get() = definedExternally\n  set(value) =
definedExternally\n  var integrity: String?\n  get() = definedExternally\n  set(value) = definedExternally\n
var keepalive: Boolean?\n  get() = definedExternally\n  set(value) = definedExternally\n  var window:
Any?\n  get() = definedExternally\n  set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun RequestInit(method: String? =
undefined, headers: dynamic = undefined, body: dynamic = undefined, referrer: String? = undefined, referrerPolicy:
dynamic = undefined, mode: RequestMode? = undefined, credentials: RequestCredentials? = undefined, cache:
RequestCache? = undefined, redirect: RequestRedirect? = undefined, integrity: String? = undefined, keepalive:
Boolean? = undefined, window: Any? = undefined): RequestInit {\n  val o = js(\"({})\")\n  o[\"method\"] =
method\n  o[\"headers\"] = headers\n  o[\"body\"] = body\n  o[\"referrer\"] = referrer\n  o[\"referrerPolicy\"] =
referrerPolicy\n  o[\"mode\"] = mode\n  o[\"credentials\"] = credentials\n  o[\"cache\"] = cache\n  o[\"redirect\"]
= redirect\n  o[\"integrity\"] = integrity\n  o[\"keepalive\"] = keepalive\n  o[\"window\"] = window\n  return
o\n}\n\n/**\n * Exposes the JavaScript [Response](https://developer.mozilla.org/en/docs/Web/API/Response) to
Kotlin\n */\npublic external open class Response(body: dynamic = definedExternally, init: ResponseInit =
definedExternally) : Body {\n  open val type: ResponseType\n  open val url: String\n  open val redirected:
Boolean\n  open val status: Short\n  open val ok: Boolean\n  open val statusText: String\n  open val headers:
Headers\n  open val body: dynamic\n  open val trailer: Promise<Headers>\n  override val bodyUsed: Boolean\n
fun clone(): Response\n  override fun arrayBuffer(): Promise<ArrayBuffer>\n  override fun blob():
Promise<Blob>\n  override fun formData(): Promise<FormData>\n  override fun json(): Promise<Any?>\n
override fun text(): Promise<String>\n\n  companion object {\n    fun error(): Response\n    fun redirect(url:
String, status: Short = definedExternally): Response\n  }\n}\n\npublic external interface ResponseInit {\n  var
status: Short? /* = 200 */\n  get() = definedExternally\n  set(value) = definedExternally\n  var statusText:
String? /* = \"OK\" */\n  get() = definedExternally\n  set(value) = definedExternally\n  var headers:
dynamic\n  get() = definedExternally\n  set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun ResponseInit(status: Short? = 200,
statusText: String? = \"OK\", headers: dynamic = undefined): ResponseInit {\n  val o = js(\"({})\")\n  o[\"status\"]
= status\n  o[\"statusText\"] = statusText\n  o[\"headers\"] = headers\n  return o\n}\n\n/* please, don't implement
this interface! */\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\npublic
external interface RequestType {\n  companion object\n}\n\npublic inline val RequestType.Companion.EMPTY:
RequestType get() = \"\".asDynamic().unsafeCast<RequestType>()\n\npublic inline val
RequestType.Companion.AUDIO: RequestType get() =
\"audio\".asDynamic().unsafeCast<RequestType>()\n\npublic inline val RequestType.Companion.FONT:
RequestType get() = \"font\".asDynamic().unsafeCast<RequestType>()\n\npublic inline val
RequestType.Companion.IMAGE: RequestType get() =
\"image\".asDynamic().unsafeCast<RequestType>()\n\npublic inline val RequestType.Companion.SCRIPT:

```

```

RequestType get() = `script`.asDynamic().unsafeCast<RequestType>()\n\npublic inline val
RequestType.Companion.STYLE: RequestType get() =
`style`.asDynamic().unsafeCast<RequestType>()\n\npublic inline val RequestType.Companion.TRACK:
RequestType get() = `track`.asDynamic().unsafeCast<RequestType>()\n\npublic inline val
RequestType.Companion.VIDEO: RequestType get() = `video`.asDynamic().unsafeCast<RequestType>()\n\n/*
please, don't implement this interface!
*/\n\n@JsName(`null`)\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic external
interface RequestDestination {\n    companion object\n}\n\npublic inline val
RequestDestination.Companion.EMPTY: RequestDestination get() =
``.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.DOCUMENT: RequestDestination get() =
`document`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.EMBED: RequestDestination get() =
`embed`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.FONT: RequestDestination get() =
`font`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.IMAGE: RequestDestination get() =
`image`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.MANIFEST: RequestDestination get() =
`manifest`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.MEDIA: RequestDestination get() =
`media`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.OBJECT: RequestDestination get() =
`object`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.REPORT: RequestDestination get() =
`report`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.SCRIPT: RequestDestination get() =
`script`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.SERVICEWORKER: RequestDestination get() =
`serviceworker`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.SHAREDWORKER: RequestDestination get() =
`sharedworker`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.STYLE: RequestDestination get() =
`style`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.WORKER: RequestDestination get() =
`worker`.asDynamic().unsafeCast<RequestDestination>()\n\npublic inline val
RequestDestination.Companion.XSLT: RequestDestination get() =
`xslt`.asDynamic().unsafeCast<RequestDestination>()\n\n/* please, don't implement this interface!
*/\n\n@JsName(`null`)\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic external
interface RequestMode {\n    companion object\n}\n\npublic inline val RequestMode.Companion.NAVIGATE:
RequestMode get() = `navigate`.asDynamic().unsafeCast<RequestMode>()\n\npublic inline val
RequestMode.Companion.SAME_ORIGIN: RequestMode get() = `same-
origin`.asDynamic().unsafeCast<RequestMode>()\n\npublic inline val RequestMode.Companion.NO_CORS:
RequestMode get() = `no-cors`.asDynamic().unsafeCast<RequestMode>()\n\npublic inline val
RequestMode.Companion.CORS: RequestMode get() = `cors`.asDynamic().unsafeCast<RequestMode>()\n\n/*
please, don't implement this interface!
*/\n\n@JsName(`null`)\n@Suppress(`NESTED_CLASS_IN_EXTERNAL_INTERFACE`)\n\npublic external
interface RequestCredentials {\n    companion object\n}\n\npublic inline val RequestCredentials.Companion.OMIT:

```

```

RequestCredentials get() = \"omit\".asDynamic().unsafeCast<RequestCredentials>()\n\npublic inline val
RequestCredentials.Companion.SAME_ORIGIN: RequestCredentials get() = \"same-
origin\".asDynamic().unsafeCast<RequestCredentials>()\n\npublic inline val
RequestCredentials.Companion.INCLUDE: RequestCredentials get() =
\"include\".asDynamic().unsafeCast<RequestCredentials>()\n\n/* please, don't implement this interface!
*\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\n\npublic external
interface RequestCache {\n    companion object\n}\n\npublic inline val RequestCache.Companion.DEFAULT:
RequestCache get() = \"default\".asDynamic().unsafeCast<RequestCache>()\n\npublic inline val
RequestCache.Companion.NO_STORE: RequestCache get() = \"no-
store\".asDynamic().unsafeCast<RequestCache>()\n\npublic inline val RequestCache.Companion.RELOAD:
RequestCache get() = \"reload\".asDynamic().unsafeCast<RequestCache>()\n\npublic inline val
RequestCache.Companion.NO_CACHE: RequestCache get() = \"no-
cache\".asDynamic().unsafeCast<RequestCache>()\n\npublic inline val
RequestCache.Companion.FORCE_CACHE: RequestCache get() = \"force-
cache\".asDynamic().unsafeCast<RequestCache>()\n\npublic inline val
RequestCache.Companion.ONLY_IF_CACHED: RequestCache get() = \"only-if-
cached\".asDynamic().unsafeCast<RequestCache>()\n\n/* please, don't implement this interface!
*\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\n\npublic external
interface RequestRedirect {\n    companion object\n}\n\npublic inline val RequestRedirect.Companion.FOLLOW:
RequestRedirect get() = \"follow\".asDynamic().unsafeCast<RequestRedirect>()\n\npublic inline val
RequestRedirect.Companion.ERROR: RequestRedirect get() =
\"error\".asDynamic().unsafeCast<RequestRedirect>()\n\npublic inline val RequestRedirect.Companion.MANUAL:
RequestRedirect get() = \"manual\".asDynamic().unsafeCast<RequestRedirect>()\n\n/* please, don't implement this
interface! *\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\n\npublic
external interface ResponseType {\n    companion object\n}\n\npublic inline val ResponseType.Companion.BASIC:
ResponseType get() = \"basic\".asDynamic().unsafeCast<ResponseType>()\n\npublic inline val
ResponseType.Companion.CORS: ResponseType get() =
\"cors\".asDynamic().unsafeCast<ResponseType>()\n\npublic inline val ResponseType.Companion.DEFAULT:
ResponseType get() = \"default\".asDynamic().unsafeCast<ResponseType>()\n\npublic inline val
ResponseType.Companion.ERROR: ResponseType get() =
\"error\".asDynamic().unsafeCast<ResponseType>()\n\npublic inline val ResponseType.Companion.OPAQUE:
ResponseType get() = \"opaque\".asDynamic().unsafeCast<ResponseType>()\n\npublic inline val
ResponseType.Companion.OPAQUEREDIRECT: ResponseType get() =
\"opaqueredirect\".asDynamic().unsafeCast<ResponseType>()\n\n/*\n * Copyright 2010-2021 JetBrains s.r.o. and
Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n *\n@n// NOTE: THIS FILE IS AUTO-GENERATED, DO NOT
EDIT!\n// See github.com/kotlin/dukat for details\n\npackage org.w3c.dom.mediacapture\n\nimport
kotlin.js.*\nimport org.khronos.webgl.*\nimport org.w3c.dom.*\nimport org.w3c.dom.events.*\n\n/**\n * Exposes
the JavaScript [MediaStream](https://developer.mozilla.org/en/docs/Web/API/MediaStream) to Kotlin\n *\n\npublic
external open class MediaStream() : EventTarget, MediaProvider {\n    constructor(stream: MediaStream)\n
    constructor(tracks: Array<MediaStreamTrack>)\n    open val id: String\n    open val active: Boolean\n    var
onaddtrack: ((MediaStreamTrackEvent) -> dynamic)?\n    var onremovetrack: ((MediaStreamTrackEvent) ->
dynamic)?\n    fun getAudioTracks(): Array<MediaStreamTrack>\n    fun getVideoTracks():
Array<MediaStreamTrack>\n    fun getTracks(): Array<MediaStreamTrack>\n    fun getTrackById(trackId: String):
MediaStreamTrack?\n    fun addTrack(track: MediaStreamTrack)\n    fun removeTrack(track: MediaStreamTrack)\n
    fun clone(): MediaStream\n}\n\n/**\n * Exposes the JavaScript
[MediaStreamTrack](https://developer.mozilla.org/en/docs/Web/API/MediaStreamTrack) to Kotlin\n *\n\npublic
external abstract class MediaStreamTrack : EventTarget {\n    open val kind: String\n    open val id: String\n    open

```

```

val label: String\n    open var enabled: Boolean\n    open val muted: Boolean\n    open var onmute: ((Event) ->
dynamic)?\n    open var onunmute: ((Event) -> dynamic)?\n    open val readyState: MediaStreamTrackState\n
open var onended: ((Event) -> dynamic)?\n    open var onoverconstrained: ((Event) -> dynamic)?\n    fun clone():
MediaStreamTrack\n    fun stop()\n    fun getCapabilities(): MediaTrackCapabilities\n    fun getConstraints():
MediaTrackConstraints\n    fun getSettings(): MediaTrackSettings\n    fun applyConstraints(constraints:
MediaTrackConstraints = definedExternally): Promise<Unit>\n}\n\n/**\n * Exposes the JavaScript
[MediaTrackSupportedConstraints](https://developer.mozilla.org/en/docs/Web/API/MediaTrackSupportedConstrain
ts) to Kotlin\n */\n\npublic external interface MediaTrackSupportedConstraints {\n    var width: Boolean? /* = true
*/\n    get() = definedExternally\n    set(value) = definedExternally\n    var height: Boolean? /* = true */\n
get() = definedExternally\n    set(value) = definedExternally\n    var aspectRatio: Boolean? /* = true */\n    get()
= definedExternally\n    set(value) = definedExternally\n    var frameRate: Boolean? /* = true */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var facingMode: Boolean? /* = true */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var resizeMode: Boolean? /* = true */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var volume: Boolean? /* = true */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var sampleRate: Boolean? /* = true */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var sampleSize: Boolean? /* = true */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var echoCancellation: Boolean? /* = true */\n    get()
= definedExternally\n    set(value) = definedExternally\n    var autoGainControl: Boolean? /* = true */\n    get()
= definedExternally\n    set(value) = definedExternally\n    var noiseSuppression: Boolean? /* = true */\n
get() = definedExternally\n    set(value) = definedExternally\n    var latency: Boolean? /* = true */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var channelCount: Boolean? /* = true */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var deviceId: Boolean? /* = true */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var groupId: Boolean? /* = true */\n    get() =
definedExternally\n    set(value) = definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun
MediaTrackSupportedConstraints(width: Boolean? = true, height: Boolean? = true, aspectRatio: Boolean? = true,
frameRate: Boolean? = true, facingMode: Boolean? = true, resizeMode: Boolean? = true, volume: Boolean? = true,
sampleRate: Boolean? = true, sampleSize: Boolean? = true, echoCancellation: Boolean? = true, autoGainControl:
Boolean? = true, noiseSuppression: Boolean? = true, latency: Boolean? = true, channelCount: Boolean? = true,
deviceId: Boolean? = true, groupId: Boolean? = true): MediaTrackSupportedConstraints {\n    val o = js(\"({})\")\n
o[\"width\"] = width\n    o[\"height\"] = height\n    o[\"aspectRatio\"] = aspectRatio\n    o[\"frameRate\"] =
frameRate\n    o[\"facingMode\"] = facingMode\n    o[\"resizeMode\"] = resizeMode\n    o[\"volume\"] = volume\n
o[\"sampleRate\"] = sampleRate\n    o[\"sampleSize\"] = sampleSize\n    o[\"echoCancellation\"] =
echoCancellation\n    o[\"autoGainControl\"] = autoGainControl\n    o[\"noiseSuppression\"] = noiseSuppression\n
o[\"latency\"] = latency\n    o[\"channelCount\"] = channelCount\n    o[\"deviceId\"] = deviceId\n    o[\"groupId\"] =
groupId\n    return o\n}\n\npublic external interface MediaTrackCapabilities {\n    var width: ULongRange?\n
get() = definedExternally\n    set(value) = definedExternally\n    var height: ULongRange?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var aspectRatio: DoubleRange?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var frameRate: DoubleRange?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var facingMode: Array<String>?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var resizeMode: Array<String>?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var volume: DoubleRange?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var sampleRate: ULongRange?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var sampleSize: ULongRange?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var echoCancellation: Array<Boolean>?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var autoGainControl: Array<Boolean>?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var noiseSuppression: Array<Boolean>?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var latency: DoubleRange?\n    get() =

```

```

definedExternally\n    set(value) = definedExternally\n    var channelCount: ULongRange?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var deviceId: String?\n    get() = definedExternally\n
    set(value) = definedExternally\n    var groupId: String?\n    get() = definedExternally\n    set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun MediaTrackCapabilities(width:
ULongRange? = undefined, height: ULongRange? = undefined, aspectRatio: DoubleRange? = undefined,
frameRate: DoubleRange? = undefined, facingMode: Array<String>? = undefined, resizeMode: Array<String>? =
undefined, volume: DoubleRange? = undefined, sampleRate: ULongRange? = undefined, sampleSize:
ULongRange? = undefined, echoCancellation: Array<Boolean>? = undefined, autoGainControl: Array<Boolean>?
= undefined, noiseSuppression: Array<Boolean>? = undefined, latency: DoubleRange? = undefined, channelCount:
ULongRange? = undefined, deviceId: String? = undefined, groupId: String? = undefined): MediaTrackCapabilities
{\n    val o = js(\"({})\")\n    o[\"width\"] = width\n    o[\"height\"] = height\n    o[\"aspectRatio\"] = aspectRatio\n
    o[\"frameRate\"] = frameRate\n    o[\"facingMode\"] = facingMode\n    o[\"resizeMode\"] = resizeMode\n
    o[\"volume\"] = volume\n    o[\"sampleRate\"] = sampleRate\n    o[\"sampleSize\"] = sampleSize\n
    o[\"echoCancellation\"] = echoCancellation\n    o[\"autoGainControl\"] = autoGainControl\n
    o[\"noiseSuppression\"] = noiseSuppression\n    o[\"latency\"] = latency\n    o[\"channelCount\"] = channelCount\n
    o[\"deviceId\"] = deviceId\n    o[\"groupId\"] = groupId\n    return o\n}\n\n/**\n * Exposes the JavaScript
[MediaTrackConstraints](https://developer.mozilla.org/en/docs/Web/API/MediaTrackConstraints) to Kotlin\n
*\n\npublic external interface MediaTrackConstraints : MediaTrackConstraintSet {\n    var advanced:
Array<MediaTrackConstraintSet>?\n    get() = definedExternally\n    set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun MediaTrackConstraints(advanced:
Array<MediaTrackConstraintSet>? = undefined, width: dynamic = undefined, height: dynamic = undefined,
aspectRatio: dynamic = undefined, frameRate: dynamic = undefined, facingMode: dynamic = undefined,
resizeMode: dynamic = undefined, volume: dynamic = undefined, sampleRate: dynamic = undefined, sampleSize:
dynamic = undefined, echoCancellation: dynamic = undefined, autoGainControl: dynamic = undefined,
noiseSuppression: dynamic = undefined, latency: dynamic = undefined, channelCount: dynamic = undefined,
deviceId: dynamic = undefined, groupId: dynamic = undefined): MediaTrackConstraints {\n    val o = js(\"({})\")\n
    o[\"advanced\"] = advanced\n    o[\"width\"] = width\n    o[\"height\"] = height\n    o[\"aspectRatio\"] =
aspectRatio\n    o[\"frameRate\"] = frameRate\n    o[\"facingMode\"] = facingMode\n    o[\"resizeMode\"] =
resizeMode\n    o[\"volume\"] = volume\n    o[\"sampleRate\"] = sampleRate\n    o[\"sampleSize\"] = sampleSize\n
    o[\"echoCancellation\"] = echoCancellation\n    o[\"autoGainControl\"] = autoGainControl\n
    o[\"noiseSuppression\"] = noiseSuppression\n    o[\"latency\"] = latency\n    o[\"channelCount\"] = channelCount\n
    o[\"deviceId\"] = deviceId\n    o[\"groupId\"] = groupId\n    return o\n}\n\npublic external interface
MediaTrackConstraintSet {\n    var width: dynamic\n    get() = definedExternally\n    set(value) =
definedExternally\n    var height: dynamic\n    get() = definedExternally\n    set(value) = definedExternally\n
    var aspectRatio: dynamic\n    get() = definedExternally\n    set(value) = definedExternally\n    var frameRate:
dynamic\n    get() = definedExternally\n    set(value) = definedExternally\n    var facingMode: dynamic\n
    get() = definedExternally\n    set(value) = definedExternally\n    var resizeMode: dynamic\n    get() =
definedExternally\n    set(value) = definedExternally\n    var volume: dynamic\n    get() = definedExternally\n
    set(value) = definedExternally\n    var sampleRate: dynamic\n    get() = definedExternally\n    set(value) =
definedExternally\n    var sampleSize: dynamic\n    get() = definedExternally\n    set(value) =
definedExternally\n    var echoCancellation: dynamic\n    get() = definedExternally\n    set(value) =
definedExternally\n    var autoGainControl: dynamic\n    get() = definedExternally\n    set(value) =
definedExternally\n    var noiseSuppression: dynamic\n    get() = definedExternally\n    set(value) =
definedExternally\n    var latency: dynamic\n    get() = definedExternally\n    set(value) = definedExternally\n
    var channelCount: dynamic\n    get() = definedExternally\n    set(value) = definedExternally\n    var deviceId:
dynamic\n    get() = definedExternally\n    set(value) = definedExternally\n    var groupId: dynamic\n    get()

```

```

= definedExternally\n    set(value) = definedExternally\n}\n\n@Suppress(\\"INVISIBLE_REFERENCE\",
\\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun MediaTrackConstraintSet(width:
dynamic = undefined, height: dynamic = undefined, aspectRatio: dynamic = undefined, frameRate: dynamic =
undefined, facingMode: dynamic = undefined, resizeMode: dynamic = undefined, volume: dynamic = undefined,
sampleRate: dynamic = undefined, sampleSize: dynamic = undefined, echoCancellation: dynamic = undefined,
autoGainControl: dynamic = undefined, noiseSuppression: dynamic = undefined, latency: dynamic = undefined,
channelCount: dynamic = undefined, deviceId: dynamic = undefined, groupId: dynamic = undefined):
MediaTrackConstraintSet {\n    val o = js(\\"({})\")\n    o[\"width\"] = width\n    o[\"height\"] = height\n
o[\"aspectRatio\"] = aspectRatio\n    o[\"frameRate\"] = frameRate\n    o[\"facingMode\"] = facingMode\n
o[\"resizeMode\"] = resizeMode\n    o[\"volume\"] = volume\n    o[\"sampleRate\"] = sampleRate\n
o[\"sampleSize\"] = sampleSize\n    o[\"echoCancellation\"] = echoCancellation\n    o[\"autoGainControl\"] =
autoGainControl\n    o[\"noiseSuppression\"] = noiseSuppression\n    o[\"latency\"] = latency\n
o[\"channelCount\"] = channelCount\n    o[\"deviceId\"] = deviceId\n    o[\"groupId\"] = groupId\n    return
o\n}\n\n/**\n * Exposes the JavaScript
[MediaTrackSettings](https://developer.mozilla.org/en/docs/Web/API/MediaTrackSettings) to Kotlin\n *\npublic
external interface MediaTrackSettings {\n    var width: Int?\n        get() = definedExternally\n        set(value) =
definedExternally\n    var height: Int?\n        get() = definedExternally\n        set(value) = definedExternally\n
    var aspectRatio: Double?\n        get() = definedExternally\n        set(value) = definedExternally\n    var
frameRate: Double?\n        get() = definedExternally\n        set(value) = definedExternally\n    var facingMode:
String?\n        get() = definedExternally\n        set(value) = definedExternally\n    var resizeMode: String?\n
        get() =
definedExternally\n        set(value) = definedExternally\n    var volume: Double?\n        get() = definedExternally\n
        set(value) = definedExternally\n    var sampleRate: Int?\n        get() = definedExternally\n        set(value) =
definedExternally\n    var sampleSize: Int?\n        get() = definedExternally\n        set(value) = definedExternally\n
    var echoCancellation: Boolean?\n        get() = definedExternally\n        set(value) = definedExternally\n    var
autoGainControl: Boolean?\n        get() = definedExternally\n        set(value) = definedExternally\n    var
noiseSuppression: Boolean?\n        get() = definedExternally\n        set(value) = definedExternally\n    var
latency: Double?\n        get() = definedExternally\n        set(value) = definedExternally\n    var channelCount:
Int?\n        get() = definedExternally\n        set(value) = definedExternally\n    var deviceId: String?\n        get() =
definedExternally\n        set(value) = definedExternally\n    var groupId: String?\n        get() = definedExternally\n
        set(value) = definedExternally\n}\n\n@Suppress(\\"INVISIBLE_REFERENCE\",
\\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun MediaTrackSettings(width: Int? =
undefined, height: Int? = undefined, aspectRatio: Double? = undefined, frameRate: Double? = undefined,
facingMode: String? = undefined, resizeMode: String? = undefined, volume: Double? = undefined, sampleRate: Int? =
undefined, sampleSize: Int? = undefined, echoCancellation: Boolean? = undefined, autoGainControl: Boolean? =
undefined, noiseSuppression: Boolean? = undefined, latency: Double? = undefined, channelCount: Int? = undefined,
deviceId: String? = undefined, groupId: String? = undefined): MediaTrackSettings {\n    val o = js(\\"({})\")\n
o[\"width\"] = width\n    o[\"height\"] = height\n    o[\"aspectRatio\"] = aspectRatio\n    o[\"frameRate\"] =
frameRate\n    o[\"facingMode\"] = facingMode\n    o[\"resizeMode\"] = resizeMode\n    o[\"volume\"] = volume\n
o[\"sampleRate\"] = sampleRate\n    o[\"sampleSize\"] = sampleSize\n    o[\"echoCancellation\"] =
echoCancellation\n    o[\"autoGainControl\"] = autoGainControl\n    o[\"noiseSuppression\"] = noiseSuppression\n
o[\"latency\"] = latency\n    o[\"channelCount\"] = channelCount\n    o[\"deviceId\"] = deviceId\n    o[\"groupId\"] =
groupId\n    return o\n}\n\n/**\n * Exposes the JavaScript
[MediaStreamTrackEvent](https://developer.mozilla.org/en/docs/Web/API/MediaStreamTrackEvent) to Kotlin\n *\npublic
external open class MediaStreamTrackEvent(type: String, eventInitDict: MediaStreamTrackEventInit) :
Event {\n    open val track: MediaStreamTrack\n\n    companion object {\n        val NONE: Short\n        val
CAPTURING_PHASE: Short\n        val AT_TARGET: Short\n        val BUBBLING_PHASE: Short\n
    }\n\n    public external interface MediaStreamTrackEventInit : EventInit {\n        var track:
MediaStreamTrack?\n    }\n\n    @Suppress(\\"INVISIBLE_REFERENCE\",

```

```

\`INVISIBLE_MEMBER\`)@kotlin.internal.InlineOnly\npublic inline fun MediaStreamTrackEventInit(track:
MediaStreamTrack?, bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false):
MediaStreamTrackEventInit {\n    val o = js("{}")\n    o["track"] = track\n    o["bubbles"] = bubbles\n
o["cancelable"] = cancelable\n    o["composed"] = composed\n    return o\n}\n\npublic external open class
OverconstrainedErrorEvent(type: String, eventInitDict: OverconstrainedErrorEventInit) : Event {\n    open val error:
dynamic\n\n    companion object {\n        val NONE: Short\n        val CAPTURING_PHASE: Short\n        val
AT_TARGET: Short\n        val BUBBLING_PHASE: Short\n    }\n}\n\npublic external interface
OverconstrainedErrorEventInit : EventInit {\n    var error: dynamic /* = null */\n    get() = definedExternally\n
set(value) = definedExternally\n}\n\n@Suppress("`INVISIBLE_REFERENCE`",
\`INVISIBLE_MEMBER\`)@kotlin.internal.InlineOnly\npublic inline fun OverconstrainedErrorEventInit(error:
dynamic = null, bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false):
OverconstrainedErrorEventInit {\n    val o = js("{}")\n    o["error"] = error\n    o["bubbles"] = bubbles\n
o["cancelable"] = cancelable\n    o["composed"] = composed\n    return o\n}\n\n/**\n * Exposes the JavaScript
[MediaDevices](https://developer.mozilla.org/en/docs/Web/API/MediaDevices) to Kotlin\n */\npublic external
abstract class MediaDevices : EventTarget {\n    open var ondevicechange: ((Event) -> dynamic)?\n    fun
enumerateDevices(): Promise<Array<MediaDeviceInfo>>\n    fun getSupportedConstraints():
MediaTrackSupportedConstraints\n    fun getUserMedia(constraints: MediaStreamConstraints = definedExternally):
Promise<MediaStream>\n}\n\n/**\n * Exposes the JavaScript
[MediaDeviceInfo](https://developer.mozilla.org/en/docs/Web/API/MediaDeviceInfo) to Kotlin\n */\npublic
external abstract class MediaDeviceInfo {\n    open val deviceId: String\n    open val kind: MediaDeviceKind\n
open val label: String\n    open val groupId: String\n    fun toJSON(): dynamic\n}\n\npublic external abstract class
InputDeviceInfo : MediaDeviceInfo {\n    fun getCapabilities(): MediaTrackCapabilities\n}\n\n/**\n * Exposes the
JavaScript [MediaStreamConstraints](https://developer.mozilla.org/en/docs/Web/API/MediaStreamConstraints) to
Kotlin\n */\npublic external interface MediaStreamConstraints {\n    var video: dynamic /* = false */\n    get() =
definedExternally\n    set(value) = definedExternally\n    var audio: dynamic /* = false */\n    get() =
definedExternally\n    set(value) = definedExternally\n}\n\n@Suppress("`INVISIBLE_REFERENCE`",
\`INVISIBLE_MEMBER\`)@kotlin.internal.InlineOnly\npublic inline fun MediaStreamConstraints(video:
dynamic = false, audio: dynamic = false): MediaStreamConstraints {\n    val o = js("{}")\n    o["video"] =
video\n    o["audio"] = audio\n    return o\n}\n\npublic external interface ConstrainingPattern {\n    var
onoverconstrained: ((Event) -> dynamic)?\n    get() = definedExternally\n    set(value) = definedExternally\n
fun getCapabilities(): Capabilities\n    fun getConstraints(): Constraints\n    fun getSettings(): Settings\n    fun
applyConstraints(constraints: Constraints = definedExternally): Promise<Unit>\n}\n\n/**\n * Exposes the
JavaScript [DoubleRange](https://developer.mozilla.org/en/docs/Web/API/DoubleRange) to Kotlin\n */\npublic
external interface DoubleRange {\n    var max: Double?\n    get() = definedExternally\n    set(value) =
definedExternally\n    var min: Double?\n    get() = definedExternally\n    set(value) =
definedExternally\n}\n\n@Suppress("`INVISIBLE_REFERENCE`",
\`INVISIBLE_MEMBER\`)@kotlin.internal.InlineOnly\npublic inline fun DoubleRange(max: Double? =
undefined, min: Double? = undefined): DoubleRange {\n    val o = js("{}")\n    o["max"] = max\n    o["min"] =
min\n    return o\n}\n\npublic external interface ConstrainDoubleRange : DoubleRange {\n    var exact: Double?\n
get() = definedExternally\n    set(value) = definedExternally\n    var ideal: Double?\n    get() =
definedExternally\n    set(value) = definedExternally\n}\n\n@Suppress("`INVISIBLE_REFERENCE`",
\`INVISIBLE_MEMBER\`)@kotlin.internal.InlineOnly\npublic inline fun ConstrainDoubleRange(exact: Double?
= undefined, ideal: Double? = undefined, max: Double? = undefined, min: Double? = undefined):
ConstrainDoubleRange {\n    val o = js("{}")\n    o["exact"] = exact\n    o["ideal"] = ideal\n    o["max"] =
max\n    o["min"] = min\n    return o\n}\n\npublic external interface ULongRange {\n    var max: Int?\n    get() =
definedExternally\n    set(value) = definedExternally\n    var min: Int?\n    get() = definedExternally\n
set(value) = definedExternally\n}\n\n@Suppress("`INVISIBLE_REFERENCE`",
\`INVISIBLE_MEMBER\`)@kotlin.internal.InlineOnly\npublic inline fun ULongRange(max: Int? = undefined,

```

```

min: Int? = undefined): ULongRange {\n  val o = js("{}")\n  o["max"] = max\n  o["min"] = min\n  return o\n}\n\npublic external interface ConstrainULongRange : ULongRange {\n  var exact: Int?\n  get() = definedExternally\n  set(value) = definedExternally\n  var ideal: Int?\n  get() = definedExternally\n  set(value) = definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",  
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun ConstrainULongRange(exact: Int? =  
undefined, ideal: Int? = undefined, max: Int? = undefined, min: Int? = undefined): ConstrainULongRange {\n  val o  
= js("{}")\n  o["exact"] = exact\n  o["ideal"] = ideal\n  o["max"] = max\n  o["min"] = min\n  return o\n}\n\n/**\n * Exposes the JavaScript  
[ConstrainBooleanParameters](https://developer.mozilla.org/en/docs/Web/API/ConstrainBooleanParameters) to  
Kotlin\n */\npublic external interface ConstrainBooleanParameters {\n  var exact: Boolean?\n  get() =  
definedExternally\n  set(value) = definedExternally\n  var ideal: Boolean?\n  get() = definedExternally\n  set(value) = definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",  
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun ConstrainBooleanParameters(exact:  
Boolean? = undefined, ideal: Boolean? = undefined): ConstrainBooleanParameters {\n  val o = js("{}")\n  o["exact"] = exact\n  o["ideal"] = ideal\n  return o\n}\n\n/**\n * Exposes the JavaScript  
[ConstrainDOMStringParameters](https://developer.mozilla.org/en/docs/Web/API/ConstrainDOMStringParameters)  
to Kotlin\n */\npublic external interface ConstrainDOMStringParameters {\n  var exact: dynamic\n  get() =  
definedExternally\n  set(value) = definedExternally\n  var ideal: dynamic\n  get() = definedExternally\n  set(value) = definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",  
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun  
ConstrainDOMStringParameters(exact: dynamic = undefined, ideal: dynamic = undefined):  
ConstrainDOMStringParameters {\n  val o = js("{}")\n  o["exact"] = exact\n  o["ideal"] = ideal\n  return o\n}\n\npublic external interface Capabilities\n\n@Suppress("INVISIBLE_REFERENCE",  
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun Capabilities(): Capabilities {\n  val o  
= js("{}")\n  return o\n}\n\npublic external interface Settings\n\n@Suppress("INVISIBLE_REFERENCE",  
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun Settings(): Settings {\n  val o =  
js("{}")\n  return o\n}\n\npublic external interface ConstraintSet\n\n@Suppress("INVISIBLE_REFERENCE",  
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun ConstraintSet(): ConstraintSet {\n  val o = js("{}")\n  return o\n}\n\npublic external interface Constraints : ConstraintSet {\n  var advanced:  
Array<ConstraintSet>?\n  get() = definedExternally\n  set(value) =  
definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",  
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun Constraints(advanced:  
Array<ConstraintSet>? = undefined): Constraints {\n  val o = js("{}")\n  o["advanced"] = advanced\n  return o\n}\n\n/* please, don't implement this interface!  
*\n */\n@JsName("null")\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\npublic external  
interface MediaStreamTrackState {\n  companion object\n}\n\npublic inline val  
MediaStreamTrackState.Companion.LIVE: MediaStreamTrackState get() =  
"live".asDynamic().unsafeCast<MediaStreamTrackState>()\n\npublic inline val  
MediaStreamTrackState.Companion.ENDED: MediaStreamTrackState get() =  
"ended".asDynamic().unsafeCast<MediaStreamTrackState>()\n\n/* please, don't implement this interface!  
*\n */\n@JsName("null")\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\npublic external  
interface VideoFacingModeEnum {\n  companion object\n}\n\npublic inline val  
VideoFacingModeEnum.Companion.USER: VideoFacingModeEnum get() =  
"user".asDynamic().unsafeCast<VideoFacingModeEnum>()\n\npublic inline val  
VideoFacingModeEnum.Companion.ENVIRONMENT: VideoFacingModeEnum get() =  
"environment".asDynamic().unsafeCast<VideoFacingModeEnum>()\n\npublic inline val  
VideoFacingModeEnum.Companion.LEFT: VideoFacingModeEnum get() =  
"left".asDynamic().unsafeCast<VideoFacingModeEnum>()\n\npublic inline val

```

```

VideoFacingModeEnum.Companion.RIGHT: VideoFacingModeEnum get() =
`"right".asDynamic().unsafeCast<VideoFacingModeEnum>()\n\n/* please, don't implement this interface!
*\n@JsName("null")\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\n\npublic external
interface VideoSizeModeEnum {\n  companion object\n}\n\npublic inline val
VideoSizeModeEnum.Companion.NONE: VideoSizeModeEnum get() =
`"none".asDynamic().unsafeCast<VideoSizeModeEnum>()\n\npublic inline val
VideoSizeModeEnum.Companion.CROP_AND_SCALE: VideoSizeModeEnum get() = `crop-and-
scale".asDynamic().unsafeCast<VideoSizeModeEnum>()\n\n/* please, don't implement this interface!
*\n@JsName("null")\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\n\npublic external
interface MediaDeviceKind {\n  companion object\n}\n\npublic inline val
MediaDeviceKind.Companion.AUDIOINPUT: MediaDeviceKind get() =
`"audioinput".asDynamic().unsafeCast<MediaDeviceKind>()\n\npublic inline val
MediaDeviceKind.Companion.AUDIOOUTPUT: MediaDeviceKind get() =
`"audiooutput".asDynamic().unsafeCast<MediaDeviceKind>()\n\npublic inline val
MediaDeviceKind.Companion.VIDEOINPUT: MediaDeviceKind get() =
`"videoinput".asDynamic().unsafeCast<MediaDeviceKind>()`,`*\n * Copyright 2010-2021 JetBrains s.r.o. and
Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n */\n\n// NOTE: THIS FILE IS AUTO-GENERATED, DO NOT
EDIT!\n\n See github.com/kotlin/dukat for details\n\npackage org.w3c.dom.mediasource\n\nimport
kotlin.js.*\nimport org.khronos.webgl.*\nimport org.w3c.dom.*\nimport org.w3c.dom.events.*\n\n/**\n * Exposes
the JavaScript [MediaSource](https://developer.mozilla.org/en/docs/Web/API/MediaSource) to Kotlin\n */\n\npublic
external open class MediaSource : EventTarget, MediaProvider {\n  open val sourceBuffers: SourceBufferList\n
open val activeSourceBuffers: SourceBufferList\n  open val readyState: ReadyState\n  var duration: Double\n
var onsourceopen: ((Event) -> dynamic)?\n  var onsourceended: ((Event) -> dynamic)?\n  var onsourceclose:
((Event) -> dynamic)?\n  fun addSourceBuffer(type: String): SourceBuffer\n  fun
removeSourceBuffer(sourceBuffer: SourceBuffer)\n  fun endOfStream(error: EndOfStreamError =
definedExternally)\n  fun setLiveSeekableRange(start: Double, end: Double)\n  fun clearLiveSeekableRange()\n\n
companion object {\n  fun isTypeSupported(type: String): Boolean\n  }\n}\n\n/**\n * Exposes the JavaScript
[SourceBuffer](https://developer.mozilla.org/en/docs/Web/API/SourceBuffer) to Kotlin\n */\n\npublic external
abstract class SourceBuffer : EventTarget {\n  open var mode: AppendMode\n  open val updating: Boolean\n
open val buffered: TimeRanges\n  open val timestampOffset: Double\n  open val audioTracks: AudioTrackList\n
open val videoTracks: VideoTrackList\n  open val textTracks: TextTrackList\n  open var appendWindowStart:
Double\n  open var appendWindowEnd: Double\n  open var onupdatestart: ((Event) -> dynamic)?\n  open var
onupdate: ((Event) -> dynamic)?\n  open var onupdateend: ((Event) -> dynamic)?\n  open var onerror: ((Event) ->
dynamic)?\n  open var onabort: ((Event) -> dynamic)?\n  fun appendBuffer(data: dynamic)\n  fun abort()\n  fun
remove(start: Double, end: Double)\n}\n\n/**\n * Exposes the JavaScript
[SourceBufferList](https://developer.mozilla.org/en/docs/Web/API/SourceBufferList) to Kotlin\n */\n\npublic
external abstract class SourceBufferList : EventTarget {\n  open val length: Int\n  open var onaddsourcebuffer:
((Event) -> dynamic)?\n  open var onremovesourcebuffer: ((Event) ->
dynamic)?\n}\n\n@Suppress("INVISIBLE_REFERENCE",
`INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\n\npublic inline operator fun SourceBufferList.get(index:
Int): SourceBuffer? = asDynamic()[index]\n\n/* please, don't implement this interface!
*\n@JsName("null")\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\n\npublic external
interface ReadyState {\n  companion object\n}\n\npublic inline val ReadyState.Companion.CLOSED: ReadyState
get() = `closed".asDynamic().unsafeCast<ReadyState>()\n\npublic inline val ReadyState.Companion.OPEN:
ReadyState get() = `open".asDynamic().unsafeCast<ReadyState>()\n\npublic inline val
ReadyState.Companion.ENDED: ReadyState get() = `ended".asDynamic().unsafeCast<ReadyState>()\n\n/*
please, don't implement this interface!

```

```

*\n@JsName("null")\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\npublic external
interface EndOfStreamError {\n    companion object\n}\n\npublic inline val
EndOfStreamError.Companion.NETWORK: EndOfStreamError get() =
"network".asDynamic().unsafeCast<EndOfStreamError>()\n\npublic inline val
EndOfStreamError.Companion.DECODE: EndOfStreamError get() =
"decode".asDynamic().unsafeCast<EndOfStreamError>()\n\n/* please, don't implement this interface!
*\n@JsName("null")\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\npublic external
interface AppendMode {\n    companion object\n}\n\npublic inline val AppendMode.Companion.SEGMENTS:
AppendMode get() = "segments".asDynamic().unsafeCast<AppendMode>()\n\npublic inline val
AppendMode.Companion.SEQUENCE: AppendMode get() =
"sequence".asDynamic().unsafeCast<AppendMode>()"/**\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin
Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.\n *\n*/\n\n// NOTE: THIS FILE IS AUTO-GENERATED, DO NOT EDIT!\n\n//
See github.com/kotlin/dukat for details\n\npackage org.w3c.dom.pointerevents\n\nimport kotlin.js.*\nimport
org.khronos.webgl.*\nimport org.w3c.dom.*\nimport org.w3c.dom.events.*\n\npublic external interface
PointerEventInit : MouseEventInit {\n    var pointerId: Int? /* = 0 */\n        get() = definedExternally\n
set(value) = definedExternally\n    var width: Double? /* = 1.0 */\n        get() = definedExternally\n
set(value) = definedExternally\n    var height: Double? /* = 1.0 */\n        get() = definedExternally\n
set(value) = definedExternally\n    var pressure: Float? /* = 0f */\n        get() = definedExternally\n
set(value) = definedExternally\n    var tangentialPressure: Float? /* = 0f */\n        get() = definedExternally\n
set(value) = definedExternally\n    var tiltX: Int? /* = 0 */\n        get() = definedExternally\n
set(value) = definedExternally\n    var tiltY: Int? /* = 0 */\n        get() = definedExternally\n
set(value) = definedExternally\n    var twist: Int? /* = 0 */\n        get() = definedExternally\n
set(value) = definedExternally\n    var pointerType: String? /* = "" */\n        get() = definedExternally\n
set(value) = definedExternally\n    var isPrimary: Boolean? /* = false */\n        get() = definedExternally\n
set(value) = definedExternally\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun PointerEventInit(pointerId: Int? = 0,
width: Double? = 1.0, height: Double? = 1.0, pressure: Float? = 0f, tangentialPressure: Float? = 0f, tiltX: Int? = 0,
tiltY: Int? = 0, twist: Int? = 0, pointerType: String? = "", isPrimary: Boolean? = false, screenX: Int? = 0, screenY:
Int? = 0, clientX: Int? = 0, clientY: Int? = 0, button: Short? = 0, buttons: Short? = 0, relatedTarget: EventTarget? =
null, region: String? = null, ctrlKey: Boolean? = false, shiftKey: Boolean? = false, altKey: Boolean? = false,
metaKey: Boolean? = false, modifierAltGraph: Boolean? = false, modifierCapsLock: Boolean? = false, modifierFn:
Boolean? = false, modifierFnLock: Boolean? = false, modifierHyper: Boolean? = false, modifierNumLock:
Boolean? = false, modifierScrollLock: Boolean? = false, modifierSuper: Boolean? = false, modifierSymbol:
Boolean? = false, modifierSymbolLock: Boolean? = false, view: Window? = null, detail: Int? = 0, bubbles:
Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false): PointerEventInit {\n    val o =
js("{}")\n    o["pointerId"] = pointerId\n    o["width"] = width\n    o["height"] = height\n    o["pressure"] =
pressure\n    o["tangentialPressure"] = tangentialPressure\n    o["tiltX"] = tiltX\n    o["tiltY"] = tiltY\n
o["twist"] = twist\n    o["pointerType"] = pointerType\n    o["isPrimary"] = isPrimary\n    o["screenX"] =
screenX\n    o["screenY"] = screenY\n    o["clientX"] = clientX\n    o["clientY"] = clientY\n    o["button"] =
button\n    o["buttons"] = buttons\n    o["relatedTarget"] = relatedTarget\n    o["region"] = region\n
o["ctrlKey"] = ctrlKey\n    o["shiftKey"] = shiftKey\n    o["altKey"] = altKey\n    o["metaKey"] = metaKey\n
o["modifierAltGraph"] = modifierAltGraph\n    o["modifierCapsLock"] = modifierCapsLock\n
o["modifierFn"] = modifierFn\n    o["modifierFnLock"] = modifierFnLock\n    o["modifierHyper"] =
modifierHyper\n    o["modifierNumLock"] = modifierNumLock\n    o["modifierScrollLock"] =
modifierScrollLock\n    o["modifierSuper"] = modifierSuper\n    o["modifierSymbol"] = modifierSymbol\n
o["modifierSymbolLock"] = modifierSymbolLock\n    o["view"] = view\n    o["detail"] = detail\n
o["bubbles"] = bubbles\n    o["cancelable"] = cancelable\n    o["composed"] = composed\n    return
o\n}\n\n/**\n * Exposes the JavaScript

```

```

[PointerEvent](https://developer.mozilla.org/en/docs/Web/API/PointerEvent) to Kotlin\n */\npublic external open
class PointerEvent(type: String, eventInitDict: PointerEventInit = definedExternally) : MouseEvent {\n  open val
pointerId: Int\n  open val width: Double\n  open val height: Double\n  open val pressure: Float\n  open val
tangentialPressure: Float\n  open val tiltX: Int\n  open val tiltY: Int\n  open val twist: Int\n  open val
pointerType: String\n  open val isPrimary: Boolean\n\n  companion object {\n    val NONE: Short\n    val
CAPTURING_PHASE: Short\n    val AT_TARGET: Short\n    val BUBBLING_PHASE: Short\n  }\n}"/\n\n
* Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code
is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n// NOTE: THIS
FILE IS AUTO-GENERATED, DO NOT EDIT!\n\n// See github.com/kotlin/dukat for details\n\npackage
org.w3c.dom.svg\n\nimport kotlin.js.*\nimport org.khronos.webgl.*\nimport org.w3c.dom.*\nimport
org.w3c.dom.css.*\n\n/**\n * Exposes the JavaScript
[SVGElement](https://developer.mozilla.org/en/docs/Web/API/SVGElement) to Kotlin\n */\npublic external
abstract class SVGElement : Element, ElementCSSInlineStyle, GlobalEventHandlers, SVGElementInstance {\n
open val dataset: DOMStringMap\n  open val ownerSVGElement: SVGSVGElement?\n  open val
viewportElement: SVGElement?\n  open var tabIndex: Int\n  fun focus()\n  fun blur()\n\n  companion object
{\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n
val CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val
ENTITY_NODE: Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE:
Short\n    val DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\npublic external interface
SVGBoundingBoxOptions {\n  var fill: Boolean? /* = true */\n    get() = definedExternally\n    set(value) =
definedExternally\n  var stroke: Boolean? /* = false */\n    get() = definedExternally\n    set(value) =
definedExternally\n  var markers: Boolean? /* = false */\n    get() = definedExternally\n    set(value) =
definedExternally\n  var clipped: Boolean? /* = false */\n    get() = definedExternally\n    set(value) =
definedExternally\n}\n\n@Suppress("\nINVISIBLE_REFERENCE",
\nINVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline fun SVGBoundingBoxOptions(fill:
Boolean? = true, stroke: Boolean? = false, markers: Boolean? = false, clipped: Boolean? = false):
SVGBoundingBoxOptions {\n  val o = js("{}")\n  o["fill"] = fill\n  o["stroke"] = stroke\n  o["markers"]
= markers\n  o["clipped"] = clipped\n  return o\n}\n\n/**\n * Exposes the JavaScript
[SVGGraphicsElement](https://developer.mozilla.org/en/docs/Web/API/SVGGraphicsElement) to Kotlin\n
*/\npublic external abstract class SVGGraphicsElement : SVGElement, SVGTests {\n  open val transform:
SVGAnimatedTransformList\n  fun getBBox(options: SVGBoundingBoxOptions = definedExternally):
DOMRect\n  fun getCTM(): DOMMatrix?\n  fun getScreenCTM(): DOMMatrix?\n\n  companion object {\n
val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val
CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE:
Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[SVGGeometryElement](https://developer.mozilla.org/en/docs/Web/API/SVGGeometryElement) to Kotlin\n
*/\npublic external abstract class SVGGeometryElement : SVGGraphicsElement {\n  open val pathLength:

```

```

SVGAnimatedNumber\n fun isPointInFill(point: DOMPoint): Boolean\n fun isPointInStroke(point: DOMPoint):
Boolean\n fun getTotalLength(): Float\n fun getPointAtLength(distance: Float): DOMPoint\n\n companion
object {\n val ELEMENT_NODE: Short\n val ATTRIBUTE_NODE: Short\n val TEXT_NODE:
Short\n val CDATA_SECTION_NODE: Short\n val ENTITY_REFERENCE_NODE: Short\n val
ENTITY_NODE: Short\n val PROCESSING_INSTRUCTION_NODE: Short\n val COMMENT_NODE:
Short\n val DOCUMENT_NODE: Short\n val DOCUMENT_TYPE_NODE: Short\n val
DOCUMENT_FRAGMENT_NODE: Short\n val NOTATION_NODE: Short\n val
DOCUMENT_POSITION_DISCONNECTED: Short\n val DOCUMENT_POSITION_PRECEDING: Short\n
val DOCUMENT_POSITION_FOLLOWING: Short\n val DOCUMENT_POSITION_CONTAINS: Short\n
val DOCUMENT_POSITION_CONTAINED_BY: Short\n val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n }\n\n\n**\n * Exposes the JavaScript
[SVGNumber](https://developer.mozilla.org/en/docs/Web/API/SVGNumber) to Kotlin\n *^npublic external
abstract class SVGNumber {\n open var value: Float\n}\n\n**\n * Exposes the JavaScript
[SVGLength](https://developer.mozilla.org/en/docs/Web/API/SVGLength) to Kotlin\n *^npublic external abstract
class SVGLength {\n open val unitType: Short\n open var value: Float\n open var valueInSpecifiedUnits:
Float\n open var valueAsString: String\n fun newValueSpecifiedUnits(unitType: Short, valueInSpecifiedUnits:
Float)\n fun convertToSpecifiedUnits(unitType: Short)\n\n companion object {\n val
SVG_LENGTHTYPE_UNKNOWN: Short\n val SVG_LENGTHTYPE_NUMBER: Short\n val
SVG_LENGTHTYPE_PERCENTAGE: Short\n val SVG_LENGTHTYPE_EMS: Short\n val
SVG_LENGTHTYPE_EXS: Short\n val SVG_LENGTHTYPE_PX: Short\n val
SVG_LENGTHTYPE_CM: Short\n val SVG_LENGTHTYPE_MM: Short\n val
SVG_LENGTHTYPE_IN: Short\n val SVG_LENGTHTYPE_PT: Short\n val SVG_LENGTHTYPE_PC:
Short\n }\n}\n\n**\n * Exposes the JavaScript
[SVGAngle](https://developer.mozilla.org/en/docs/Web/API/SVGAngle) to Kotlin\n *^npublic external abstract
class SVGAngle {\n open val unitType: Short\n open var value: Float\n open var valueInSpecifiedUnits:
Float\n open var valueAsString: String\n fun newValueSpecifiedUnits(unitType: Short, valueInSpecifiedUnits:
Float)\n fun convertToSpecifiedUnits(unitType: Short)\n\n companion object {\n val
SVG_ANGLETYPE_UNKNOWN: Short\n val SVG_ANGLETYPE_UNSPECIFIED: Short\n val
SVG_ANGLETYPE_DEG: Short\n val SVG_ANGLETYPE_RAD: Short\n val
SVG_ANGLETYPE_GRAD: Short\n }\n}\n\npublic external abstract class SVGNameList {\n open val length:
Int\n open val numberOfItems: Int\n fun clear()\n fun initialize(newItem: dynamic): dynamic\n fun
insertItemBefore(newItem: dynamic, index: Int): dynamic\n fun replaceItem(newItem: dynamic, index: Int):
dynamic\n fun removeItem(index: Int): dynamic\n fun appendItem(newItem: dynamic): dynamic\n fun
getItem(index: Int): dynamic\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun SVGNameList.get(index: Int):
dynamic = asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun SVGNameList.set(index: Int,
newItem: dynamic) { asDynamic()[index] = newItem }\n\n**\n * Exposes the JavaScript
[SVGNumberList](https://developer.mozilla.org/en/docs/Web/API/SVGNumberList) to Kotlin\n *^npublic external
abstract class SVGNumberList {\n open val length: Int\n open val numberOfItems: Int\n fun clear()\n fun
initialize(newItem: SVGNumber): SVGNumber\n fun insertItemBefore(newItem: SVGNumber, index: Int):
SVGNumber\n fun replaceItem(newItem: SVGNumber, index: Int): SVGNumber\n fun removeItem(index: Int):
SVGNumber\n fun appendItem(newItem: SVGNumber): SVGNumber\n fun getItem(index: Int):
SVGNumber\n}\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun SVGNumberList.get(index:
Int): SVGNumber? = asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun SVGNumberList.set(index:
Int, newItem: SVGNumber) { asDynamic()[index] = newItem }\n\n**\n * Exposes the JavaScript

```

[SVGLengthList](https://developer.mozilla.org/en/docs/Web/API/SVGLengthList) to Kotlin\n \*^\\npublic external abstract class SVGLengthList {\n open val length: Int\n open val numberOfItems: Int\n fun clear()\n fun initialize(newItem: SVGLength): SVGLength\n fun insertItemBefore(newItem: SVGLength, index: Int): SVGLength\n fun replaceItem(newItem: SVGLength, index: Int): SVGLength\n fun removeItem(index: Int): SVGLength\n fun appendItem(newItem: SVGLength): SVGLength\n fun getItem(index: Int): SVGLength\n}\n\n@Suppress(\"INVISIBLE\_REFERENCE\", \"INVISIBLE\_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun SVGLengthList.get(index: Int): SVGLength? = asDynamic()[index]\n\n@Suppress(\"INVISIBLE\_REFERENCE\", \"INVISIBLE\_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun SVGLengthList.set(index: Int, newItem: SVGLength) { asDynamic()[index] = newItem }\n\n/\*\*\n \* Exposes the JavaScript

[SVGAnimatedBoolean](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedBoolean) to Kotlin\n \*^\\npublic external abstract class SVGAnimatedBoolean {\n open var baseVal: Boolean\n open val animVal: Boolean\n}\n\n/\*\*\n \* Exposes the JavaScript

[SVGAnimatedEnumeration](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedEnumeration) to Kotlin\n \*^\\npublic external abstract class SVGAnimatedEnumeration {\n open var baseVal: Short\n open val animVal: Short\n}\n\n/\*\*\n \* Exposes the JavaScript

[SVGAnimatedInteger](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedInteger) to Kotlin\n \*^\\npublic external abstract class SVGAnimatedInteger {\n open var baseVal: Int\n open val animVal: Int\n}\n\n/\*\*\n \* Exposes the JavaScript

[SVGAnimatedNumber](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedNumber) to Kotlin\n \*^\\npublic external abstract class SVGAnimatedNumber {\n open var baseVal: Float\n open val animVal: Float\n}\n\n/\*\*\n \* Exposes the JavaScript

[SVGAnimatedLength](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedLength) to Kotlin\n \*^\\npublic external abstract class SVGAnimatedLength {\n open val baseVal: SVGLength\n open val animVal: SVGLength\n}\n\n/\*\*\n \* Exposes the JavaScript

[SVGAnimatedAngle](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedAngle) to Kotlin\n \*^\\npublic external abstract class SVGAnimatedAngle {\n open val baseVal: SVGAngle\n open val animVal: SVGAngle\n}\n\n/\*\*\n \* Exposes the JavaScript

[SVGAnimatedString](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedString) to Kotlin\n \*^\\npublic external abstract class SVGAnimatedString {\n open var baseVal: String\n open val animVal: String\n}\n\n/\*\*\n \* Exposes the JavaScript

[SVGAnimatedRect](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedRect) to Kotlin\n \*^\\npublic external abstract class SVGAnimatedRect {\n open val baseVal: DOMRect\n open val animVal: DOMRectReadOnly\n}\n\n/\*\*\n \* Exposes the JavaScript

[SVGAnimatedNumberList](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedNumberList) to Kotlin\n \*^\\npublic external abstract class SVGAnimatedNumberList {\n open val baseVal: SVGNumberList\n open val animVal: SVGNumberList\n}\n\n/\*\*\n \* Exposes the JavaScript

[SVGAnimatedLengthList](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedLengthList) to Kotlin\n \*^\\npublic external abstract class SVGAnimatedLengthList {\n open val baseVal: SVGLengthList\n open val animVal: SVGLengthList\n}\n\n/\*\*\n \* Exposes the JavaScript

[SVGStringList](https://developer.mozilla.org/en/docs/Web/API/SVGStringList) to Kotlin\n \*^\\npublic external abstract class SVGStringList {\n open val length: Int\n open val numberOfItems: Int\n fun clear()\n fun initialize(newItem: String): String\n fun insertItemBefore(newItem: String, index: Int): String\n fun replaceItem(newItem: String, index: Int): String\n fun removeItem(index: Int): String\n fun appendItem(newItem: String): String\n fun getItem(index: Int): String\n}\n\n@Suppress(\"INVISIBLE\_REFERENCE\", \"INVISIBLE\_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun SVGStringList.get(index: Int): String? = asDynamic()[index]\n\n@Suppress(\"INVISIBLE\_REFERENCE\", \"INVISIBLE\_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun SVGStringList.set(index: Int,

```

newItem: String) { asDynamic()[index] = newItem }\n\n/**\n * Exposes the JavaScript
[SVGUnitTypes](https://developer.mozilla.org/en/docs/Web/API/SVGUnitTypes) to Kotlin\n
*\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\npublic external interface SVGUnitTypes
{\n    companion object {\n        val SVG_UNIT_TYPE_UNKNOWN: Short\n        val
SVG_UNIT_TYPE_USERSPACEONUSE: Short\n        val SVG_UNIT_TYPE_OBJECTBOUNDINGBOX:
Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGTTests](https://developer.mozilla.org/en/docs/Web/API/SVGTTests) to Kotlin\n *\npublic external interface
SVGTTests {\n    val requiredExtensions: SVGStringList\n    val systemLanguage: SVGStringList\n}\n\npublic
external interface SVGFitToViewBox {\n    val viewBox: SVGAnimatedRect\n    val preserveAspectRatio:
SVGAnimatedPreserveAspectRatio\n}\n\n/**\n * Exposes the JavaScript
[SVGZoomAndPan](https://developer.mozilla.org/en/docs/Web/API/SVGZoomAndPan) to Kotlin\n
*\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\npublic external interface
SVGZoomAndPan {\n    var zoomAndPan: Short\n\n    companion object {\n        val
SVG_ZOOMANDPAN_UNKNOWN: Short\n        val SVG_ZOOMANDPAN_DISABLE: Short\n        val
SVG_ZOOMANDPAN_MAGNIFY: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGURIReference](https://developer.mozilla.org/en/docs/Web/API/SVGURIReference) to Kotlin\n *\npublic
external interface SVGURIReference {\n    val href: SVGAnimatedString\n}\n\n/**\n * Exposes the JavaScript
[SVGSVGElement](https://developer.mozilla.org/en/docs/Web/API/SVGSVGElement) to Kotlin\n *\npublic
external abstract class SVGSVGElement : SVGGraphicsElement, SVGFitToViewBox, SVGZoomAndPan,
WindowEventHandlers {\n    open val x: SVGAnimatedLength\n    open val y: SVGAnimatedLength\n    open val
width: SVGAnimatedLength\n    open val height: SVGAnimatedLength\n    open var currentScale: Float\n    open
val currentTranslate: DOMPointReadOnly\n    fun getIntersectionList(rect: DOMRectReadOnly, referenceElement:
SVGElement?): NodeList\n    fun getEnclosureList(rect: DOMRectReadOnly, referenceElement: SVGElement?):
NodeList\n    fun checkIntersection(element: SVGElement, rect: DOMRectReadOnly): Boolean\n    fun
checkEnclosure(element: SVGElement, rect: DOMRectReadOnly): Boolean\n    fun deselectAll()\n    fun
createSVGNumber(): SVGNumber\n    fun createSVGLength(): SVGLength\n    fun createSVGAngle():
SVGAngle\n    fun createSVGPoint(): DOMPoint\n    fun createSVGMatrix(): DOMMatrix\n    fun
createSVGRect(): DOMRect\n    fun createSVGTransform(): SVGTransform\n    fun
createSVGTransformFromMatrix(matrix: DOMMatrixReadOnly): SVGTransform\n    fun
getElementById(elementId: String): Element\n    fun suspendRedraw(maxWaitMilliseconds: Int): Int\n    fun
unsuspendRedraw(suspendHandleID: Int)\n    fun unsuspendRedrawAll()\n    fun forceRedraw()\n\n    companion
object {\n        val SVG_ZOOMANDPAN_UNKNOWN: Short\n        val SVG_ZOOMANDPAN_DISABLE:
Short\n        val SVG_ZOOMANDPAN_MAGNIFY: Short\n        val ELEMENT_NODE: Short\n        val
ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val
ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val
PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGGElement](https://developer.mozilla.org/en/docs/Web/API/SVGGElement) to Kotlin\n *\npublic external
abstract class SVGGElement : SVGGraphicsElement {\n    companion object {\n        val ELEMENT_NODE:
Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE:
Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val
PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val

```

```

DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\npublic external abstract class
SVGUnknownElement : SVGGraphicsElement {\n    companion object {\n        val ELEMENT_NODE: Short\n
val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n
val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val
PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGDefsElement](https://developer.mozilla.org/en/docs/Web/API/SVGDefsElement) to Kotlin\n */\npublic
external abstract class SVGDefsElement : SVGGraphicsElement {\n    companion object {\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGDescElement](https://developer.mozilla.org/en/docs/Web/API/SVGDescElement) to Kotlin\n */\npublic
external abstract class SVGDescElement : SVGElement {\n    companion object {\n        val ELEMENT_NODE:
Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE:
Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val
PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGMetadataElement](https://developer.mozilla.org/en/docs/Web/API/SVGMetadataElement) to Kotlin\n */\npublic
external abstract class SVGMetadataElement : SVGElement {\n    companion object {\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript

```

```

[SVGTitleElement](https://developer.mozilla.org/en/docs/Web/API/SVGTitleElement) to Kotlin\n *\npublic
external abstract class SVGTitleElement : SVGElement {\n  companion object {\n    val ELEMENT_NODE:
Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE:
Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[SVGSymbolElement](https://developer.mozilla.org/en/docs/Web/API/SVGSymbolElement) to Kotlin\n *\npublic
external abstract class SVGSymbolElement : SVGGraphicsElement, SVGFitToViewBox {\n  companion object
{\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n
val CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val
ENTITY_NODE: Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE:
Short\n    val DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript
[SVGUseElement](https://developer.mozilla.org/en/docs/Web/API/SVGUseElement) to Kotlin\n *\npublic external
abstract class SVGUseElement : SVGGraphicsElement, SVGURIReference {\n  open val x:
SVGAnimatedLength\n  open val y: SVGAnimatedLength\n  open val width: SVGAnimatedLength\n  open val
height: SVGAnimatedLength\n  open val instanceRoot: SVGElement?\n  open val animatedInstanceRoot:
SVGElement?\n}\n\n  companion object {\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE:
Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\npublic external open class
SVGUseElementShadowRoot : ShadowRoot {\n  companion object {\n    val ELEMENT_NODE: Short\n
val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n
val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\npublic external interface
SVGElementInstance {\n  val correspondingElement: SVGElement?\n  get() = definedExternally\n  val
correspondingUseElement: SVGUseElement?\n  get() = definedExternally\n}\n\n\npublic external open class

```

```

ShadowAnimation(source: dynamic, newTarget: dynamic) {\n  open val sourceAnimation: dynamic\n}\n\n/**\n * Exposes the JavaScript [SVGSwitchElement](https://developer.mozilla.org/en/docs/Web/API/SVGSwitchElement) to Kotlin\n */\npublic external abstract class SVGSwitchElement : SVGGraphicsElement {\n  companion object {\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\npublic external interface GetSVGDocument {\n  fun getSVGDocument(): Document\n}\n\n/**\n * Exposes the JavaScript [SVGStyleElement](https://developer.mozilla.org/en/docs/Web/API/SVGStyleElement) to Kotlin\n */\npublic external abstract class SVGStyleElement : SVGElement, LinkStyle {\n  open var type: String\n  open var media: String\n  open var title: String\n\n  companion object {\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n}\n\n/**\n * Exposes the JavaScript [SVGTransform](https://developer.mozilla.org/en/docs/Web/API/SVGTransform) to Kotlin\n */\npublic external abstract class SVGTransform {\n  open val type: Short\n  open val matrix: DOMMatrix\n  open val angle: Float\n  fun setMatrix(matrix: DOMMatrixReadOnly)\n  fun setTranslate(tx: Float, ty: Float)\n  fun setScale(sx: Float, sy: Float)\n  fun setRotate(angle: Float, cx: Float, cy: Float)\n  fun setSkewX(angle: Float)\n  fun setSkewY(angle: Float)\n\n  companion object {\n    val SVG_TRANSFORM_UNKNOWN: Short\n    val SVG_TRANSFORM_MATRIX: Short\n    val SVG_TRANSFORM_TRANSLATE: Short\n    val SVG_TRANSFORM_SCALE: Short\n    val SVG_TRANSFORM_ROTATE: Short\n    val SVG_TRANSFORM_SKEWX: Short\n    val SVG_TRANSFORM_SKEWY: Short\n  }\n}\n\n/**\n * Exposes the JavaScript [SVGTransformList](https://developer.mozilla.org/en/docs/Web/API/SVGTransformList) to Kotlin\n */\npublic external abstract class SVGTransformList {\n  open val length: Int\n  open val numberOfItems: Int\n  fun clear()\n  fun initialize(newItem: SVGTransform): SVGTransform\n  fun insertItemBefore(newItem: SVGTransform, index: Int): SVGTransform\n  fun replaceItem(newItem: SVGTransform, index: Int): SVGTransform\n  fun removeItem(index: Int): SVGTransform\n  fun appendItem(newItem: SVGTransform): SVGTransform\n  fun createSVGTransformFromMatrix(matrix: DOMMatrixReadOnly): SVGTransform\n  fun consolidate(): SVGTransform?\n  fun getItem(index: Int): SVGTransform\n}\n\n@Suppress("INVISIBLE_REFERENCE", "INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun SVGTransformList.get(index: Int): SVGTransform? = asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE", "INVISIBLE_MEMBER")\n@kotlin.internal.InlineOnly\npublic inline operator fun SVGTransformList.set(index: Int, newItem: SVGTransform) { asDynamic()[index] = newItem }\n\n/**\n * Exposes the JavaScript [SVGAnimatedTransformList](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedTransformList) to Kotlin\n */\npublic external abstract class SVGAnimatedTransformList {\n  open val baseVal: SVGTransformList\n  open val animVal: SVGTransformList\n}\n\n/**\n * Exposes the JavaScript

```

[SVGPreserveAspectRatio](https://developer.mozilla.org/en/docs/Web/API/SVGPreserveAspectRatio) to Kotlin\n
 \*  
 public external abstract class SVGPreserveAspectRatio {\n
 open var align: Short\n
 open var meetOrSlice: Short\n
 companion object {\n
 val SVG\_PRESERVEASPECTRATIO\_UNKNOWN: Short\n
 val SVG\_PRESERVEASPECTRATIO\_NONE: Short\n
 val SVG\_PRESERVEASPECTRATIO\_XMINYMIN: Short\n
 val SVG\_PRESERVEASPECTRATIO\_XMIDYMIN: Short\n
 val SVG\_PRESERVEASPECTRATIO\_XMAXYMIN: Short\n
 val SVG\_PRESERVEASPECTRATIO\_XMINYMID: Short\n
 val SVG\_PRESERVEASPECTRATIO\_XMIDYMID: Short\n
 val SVG\_PRESERVEASPECTRATIO\_XMAXYMID: Short\n
 val SVG\_PRESERVEASPECTRATIO\_XMINYMAX: Short\n
 val SVG\_PRESERVEASPECTRATIO\_XMIDYMAX: Short\n
 val SVG\_PRESERVEASPECTRATIO\_XMAXYMAX: Short\n
 val SVG\_MEETORSlice\_UNKNOWN: Short\n
 val SVG\_MEETORSlice\_MEET: Short\n
 val SVG\_MEETORSlice\_SLICE: Short\n
 }\n
 }\n
 \*  
 Exposes the JavaScript

[SVGAnimatedPreserveAspectRatio](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedPreserveAspectRatio) to Kotlin\n
 \*  
 public external abstract class SVGAnimatedPreserveAspectRatio {\n
 open val baseVal: SVGPreserveAspectRatio\n
 open val animVal: SVGPreserveAspectRatio\n
 }\n
 \*  
 Exposes the JavaScript

[SVGPathElement](https://developer.mozilla.org/en/docs/Web/API/SVGPathElement) to Kotlin\n
 \*  
 public external abstract class SVGPathElement : SVGGeometryElement {\n
 companion object {\n
 val ELEMENT\_NODE: Short\n
 val ATTRIBUTE\_NODE: Short\n
 val TEXT\_NODE: Short\n
 val CDATA\_SECTION\_NODE: Short\n
 val ENTITY\_REFERENCE\_NODE: Short\n
 val ENTITY\_NODE: Short\n
 val PROCESSING\_INSTRUCTION\_NODE: Short\n
 val COMMENT\_NODE: Short\n
 val DOCUMENT\_NODE: Short\n
 val DOCUMENT\_TYPE\_NODE: Short\n
 val DOCUMENT\_FRAGMENT\_NODE: Short\n
 val NOTATION\_NODE: Short\n
 val DOCUMENT\_POSITION\_DISCONNECTED: Short\n
 val DOCUMENT\_POSITION\_PRECEDING: Short\n
 val DOCUMENT\_POSITION\_FOLLOWING: Short\n
 val DOCUMENT\_POSITION\_CONTAINS: Short\n
 val DOCUMENT\_POSITION\_CONTAINED\_BY: Short\n
 val DOCUMENT\_POSITION\_IMPLEMENTATION\_SPECIFIC: Short\n
 }\n
 }\n
 \*  
 Exposes the JavaScript

[SVGRectElement](https://developer.mozilla.org/en/docs/Web/API/SVGRectElement) to Kotlin\n
 \*  
 public external abstract class SVGRectElement : SVGGeometryElement {\n
 open val x: SVGAnimatedLength\n
 open val y: SVGAnimatedLength\n
 open val width: SVGAnimatedLength\n
 open val height: SVGAnimatedLength\n
 open val rx: SVGAnimatedLength\n
 open val ry: SVGAnimatedLength\n
 companion object {\n
 val ELEMENT\_NODE: Short\n
 val ATTRIBUTE\_NODE: Short\n
 val TEXT\_NODE: Short\n
 val CDATA\_SECTION\_NODE: Short\n
 val ENTITY\_REFERENCE\_NODE: Short\n
 val ENTITY\_NODE: Short\n
 val PROCESSING\_INSTRUCTION\_NODE: Short\n
 val COMMENT\_NODE: Short\n
 val DOCUMENT\_NODE: Short\n
 val DOCUMENT\_TYPE\_NODE: Short\n
 val DOCUMENT\_FRAGMENT\_NODE: Short\n
 val NOTATION\_NODE: Short\n
 val DOCUMENT\_POSITION\_DISCONNECTED: Short\n
 val DOCUMENT\_POSITION\_PRECEDING: Short\n
 val DOCUMENT\_POSITION\_FOLLOWING: Short\n
 val DOCUMENT\_POSITION\_CONTAINS: Short\n
 val DOCUMENT\_POSITION\_CONTAINED\_BY: Short\n
 val DOCUMENT\_POSITION\_IMPLEMENTATION\_SPECIFIC: Short\n
 }\n
 }\n
 \*  
 Exposes the JavaScript

[SVGCircleElement](https://developer.mozilla.org/en/docs/Web/API/SVGCircleElement) to Kotlin\n
 \*  
 public external abstract class SVGCircleElement : SVGGeometryElement {\n
 open val cx: SVGAnimatedLength\n
 open val cy: SVGAnimatedLength\n
 open val r: SVGAnimatedLength\n
 companion object {\n
 val ELEMENT\_NODE: Short\n
 val ATTRIBUTE\_NODE: Short\n
 val TEXT\_NODE: Short\n
 val CDATA\_SECTION\_NODE: Short\n
 val ENTITY\_REFERENCE\_NODE: Short\n
 val ENTITY\_NODE: Short\n
 val PROCESSING\_INSTRUCTION\_NODE: Short\n
 val COMMENT\_NODE: Short\n
 val DOCUMENT\_NODE: Short\n
 val DOCUMENT\_TYPE\_NODE: Short\n
 }\n
 }\n
 \*  
 Exposes the JavaScript

```

DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGEllipseElement](https://developer.mozilla.org/en/docs/Web/API/SVGEllipseElement) to Kotlin\n */\npublic
external abstract class SVGEllipseElement : SVGGeometryElement {\n    open val cx: SVGAnimatedLength\n
open val cy: SVGAnimatedLength\n    open val rx: SVGAnimatedLength\n    open val ry: SVGAnimatedLength\n\n
companion object {\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val
TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE:
Short\n    val ENTITY_NODE: Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val
COMMENT_NODE: Short\n    val DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n
    val DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGLineElement](https://developer.mozilla.org/en/docs/Web/API/SVGLineElement) to Kotlin\n */\npublic
external abstract class SVGLineElement : SVGGeometryElement {\n    open val x1: SVGAnimatedLength\n    open
val y1: SVGAnimatedLength\n    open val x2: SVGAnimatedLength\n    open val y2: SVGAnimatedLength\n\n
companion object {\n    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val
TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE:
Short\n    val ENTITY_NODE: Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val
COMMENT_NODE: Short\n    val DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n
    val DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGMeshElement](https://developer.mozilla.org/en/docs/Web/API/SVGMeshElement) to Kotlin\n */\npublic
external abstract class SVGMeshElement : SVGGeometryElement, SVGURIReference {\n    companion object {\n
    val ELEMENT_NODE: Short\n    val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val
CDATA_SECTION_NODE: Short\n    val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE:
Short\n    val PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGAnimatedPoints](https://developer.mozilla.org/en/docs/Web/API/SVGAnimatedPoints) to Kotlin\n */\npublic
external interface SVGAnimatedPoints {\n    val points: SVGPointList\n    val animatedPoints:
SVGPointList\n}\n\npublic external abstract class SVGPointList {\n    open val length: Int\n    open val
numberOfItems: Int\n    fun clear()\n    fun initialize(newItem: DOMPoint): DOMPoint\n    fun
insertItemBefore(newItem: DOMPoint, index: Int): DOMPoint\n    fun replaceItem(newItem: DOMPoint, index:
Int): DOMPoint\n    fun removeItem(index: Int): DOMPoint\n    fun appendItem(newItem: DOMPoint):
DOMPoint\n    fun getItem(index: Int): DOMPoint\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline operator fun SVGPointList.get(index: Int):

```

```

DOMPoint? = asDynamic()[index]\n\n@Suppress("INVISIBLE_REFERENCE",
"INVISIBLE_MEMBER")\n\n@kotlin.internal.InlineOnly\n\npublic inline operator fun SVGPointList.set(index: Int,
newItem: DOMPoint) { asDynamic()[index] = newItem }\n\n/**\n * Exposes the JavaScript
[SVGPolylineElement](https://developer.mozilla.org/en/docs/Web/API/SVGPolylineElement) to Kotlin\n
*\n\npublic external abstract class SVGPolylineElement : SVGGeometryElement, SVGAnimatedPoints {\n
companion object {\n
    val ELEMENT_NODE: Short\n
    val ATTRIBUTE_NODE: Short\n
    val TEXT_NODE: Short\n
    val CDATA_SECTION_NODE: Short\n
    val ENTITY_REFERENCE_NODE: Short\n
    val ENTITY_NODE: Short\n
    val PROCESSING_INSTRUCTION_NODE: Short\n
    val COMMENT_NODE: Short\n
    val DOCUMENT_NODE: Short\n
    val DOCUMENT_TYPE_NODE: Short\n
    val DOCUMENT_FRAGMENT_NODE: Short\n
    val NOTATION_NODE: Short\n
    val DOCUMENT_POSITION_DISCONNECTED: Short\n
    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n
    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n
    val DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n
}\n\n}\n\n/**\n * Exposes the JavaScript
[SVGPolygonElement](https://developer.mozilla.org/en/docs/Web/API/SVGPolygonElement) to Kotlin\n
*\n\npublic external abstract class SVGPolygonElement : SVGGeometryElement, SVGAnimatedPoints {\n
companion object {\n
    val ELEMENT_NODE: Short\n
    val ATTRIBUTE_NODE: Short\n
    val TEXT_NODE: Short\n
    val CDATA_SECTION_NODE: Short\n
    val ENTITY_REFERENCE_NODE: Short\n
    val ENTITY_NODE: Short\n
    val PROCESSING_INSTRUCTION_NODE: Short\n
    val COMMENT_NODE: Short\n
    val DOCUMENT_NODE: Short\n
    val DOCUMENT_TYPE_NODE: Short\n
    val DOCUMENT_FRAGMENT_NODE: Short\n
    val NOTATION_NODE: Short\n
    val DOCUMENT_POSITION_DISCONNECTED: Short\n
    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n
    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n
    val DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n
}\n\n}\n\n/**\n * Exposes the JavaScript
[SVGTextContentElement](https://developer.mozilla.org/en/docs/Web/API/SVGTextContentElement) to Kotlin\n
*\n\npublic external abstract class SVGTextContentElement : SVGGraphicsElement {\n
    open val textLength: SVGAnimatedLength\n
    open val lengthAdjust: SVGAnimatedEnumeration\n
    fun getNumberOfChars(): Int\n
    fun getComputedTextLength(): Float\n
    fun getSubStringLength(charnum: Int, nchars: Int): Float\n
    fun getStartPositionOfChar(charnum: Int): DOMPoint\n
    fun getEndPositionOfChar(charnum: Int): DOMPoint\n
    fun getExtentOfChar(charnum: Int): DOMRect\n
    fun getRotationOfChar(charnum: Int): Float\n
    fun getCharNumAtPosition(point: DOMPoint): Int\n
    fun selectSubString(charnum: Int, nchars: Int)\n\n
    companion object {\n
        val LENGTHADJUST_UNKNOWN: Short\n
        val LENGTHADJUST_SPACING: Short\n
        val LENGTHADJUST_SPACINGANDGLYPHS: Short\n
        val ELEMENT_NODE: Short\n
        val ATTRIBUTE_NODE: Short\n
        val TEXT_NODE: Short\n
        val CDATA_SECTION_NODE: Short\n
        val ENTITY_REFERENCE_NODE: Short\n
        val ENTITY_NODE: Short\n
        val PROCESSING_INSTRUCTION_NODE: Short\n
        val COMMENT_NODE: Short\n
        val DOCUMENT_NODE: Short\n
        val DOCUMENT_TYPE_NODE: Short\n
        val DOCUMENT_FRAGMENT_NODE: Short\n
        val NOTATION_NODE: Short\n
        val DOCUMENT_POSITION_DISCONNECTED: Short\n
        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n
        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n
        val DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n
    }\n\n}\n\n}\n\n/**\n * Exposes the JavaScript
[SVGTextPositioningElement](https://developer.mozilla.org/en/docs/Web/API/SVGTextPositioningElement) to
Kotlin\n
*\n\npublic external abstract class SVGTextPositioningElement : SVGTextContentElement {\n
    open val x: SVGAnimatedLengthList\n
    open val y: SVGAnimatedLengthList\n
    open val dx: SVGAnimatedLengthList\n
    open val dy: SVGAnimatedLengthList\n
    open val rotate: SVGAnimatedNumberList\n\n
    companion object {\n
        val LENGTHADJUST_UNKNOWN: Short\n
        val LENGTHADJUST_SPACING: Short\n
        val

```

```

LENGTHADJUST_SPACINGANDGLYPHS: Short\n    val ELEMENT_NODE: Short\n    val
ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGTextElement](https://developer.mozilla.org/en/docs/Web/API/SVGTextElement) to Kotlin\n */\npublic
external abstract class SVGTextElement : SVGTextPositioningElement {\n    companion object {\n    val
LENGTHADJUST_UNKNOWN: Short\n    val LENGTHADJUST_SPACING: Short\n    val
LENGTHADJUST_SPACINGANDGLYPHS: Short\n    val ELEMENT_NODE: Short\n    val
ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGTSpanElement](https://developer.mozilla.org/en/docs/Web/API/SVGTSpanElement) to Kotlin\n */\npublic
external abstract class SVGTSpanElement : SVGTextPositioningElement {\n    companion object {\n    val
LENGTHADJUST_UNKNOWN: Short\n    val LENGTHADJUST_SPACING: Short\n    val
LENGTHADJUST_SPACINGANDGLYPHS: Short\n    val ELEMENT_NODE: Short\n    val
ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGTextPathElement](https://developer.mozilla.org/en/docs/Web/API/SVGTextPathElement) to Kotlin\n
*/\npublic external abstract class SVGTextPathElement : SVGTextContentElement, SVGURIReference {\n    open
val startOffset: SVGAnimatedLength\n    open val method: SVGAnimatedEnumeration\n    open val spacing:
SVGAnimatedEnumeration\n\n    companion object {\n    val TEXTPATH_METHODTYPE_UNKNOWN:
Short\n    val TEXTPATH_METHODTYPE_ALIGN: Short\n    val
TEXTPATH_METHODTYPE_STRETCH: Short\n    val TEXTPATH_SPACINGTYPE_UNKNOWN: Short\n
    val TEXTPATH_SPACINGTYPE_AUTO: Short\n    val TEXTPATH_SPACINGTYPE_EXACT: Short\n
    val LENGTHADJUST_UNKNOWN: Short\n    val LENGTHADJUST_SPACING: Short\n    val
LENGTHADJUST_SPACINGANDGLYPHS: Short\n    val ELEMENT_NODE: Short\n    val
ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val

```

```

DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGImageElement](https://developer.mozilla.org/en/docs/Web/API/SVGImageElement) to Kotlin\n */\npublic
external abstract class SVGImageElement : SVGGraphicsElement, SVGURIReference,
HTMLOrSVGImageElement {\n    open val x: SVGAnimatedLength\n    open val y: SVGAnimatedLength\n    open
val width: SVGAnimatedLength\n    open val height: SVGAnimatedLength\n    open val preserveAspectRatio:
SVGAnimatedPreserveAspectRatio\n    open var crossOrigin: String?\n\n    companion object {\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGForeignObjectElement](https://developer.mozilla.org/en/docs/Web/API/SVGForeignObjectElement) to
Kotlin\n */\npublic external abstract class SVGForeignObjectElement : SVGGraphicsElement {\n    open val x:
SVGAnimatedLength\n    open val y: SVGAnimatedLength\n    open val width: SVGAnimatedLength\n    open val
height: SVGAnimatedLength\n\n    companion object {\n        val ELEMENT_NODE: Short\n        val
ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val
ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val
PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\npublic external abstract class
SVGMarkerElement : SVGElement, SVGFitToViewBox {\n    open val refX: SVGAnimatedLength\n    open val
refY: SVGAnimatedLength\n    open val markerUnits: SVGAnimatedEnumeration\n    open val markerWidth:
SVGAnimatedLength\n    open val markerHeight: SVGAnimatedLength\n    open val orientType:
SVGAnimatedEnumeration\n    open val orientAngle: SVGAnimatedAngle\n    open var orient: String\n    fun
setOrientToAuto()\n    fun setOrientToAngle(angle: SVGAngle)\n\n    companion object {\n        val
SVG_MARKERUNITS_UNKNOWN: Short\n        val SVG_MARKERUNITS_USERSPACEONUSE: Short\n
        val SVG_MARKERUNITS_STROKEWIDTH: Short\n        val SVG_MARKER_ORIENT_UNKNOWN: Short\n
        val SVG_MARKER_ORIENT_AUTO: Short\n        val SVG_MARKER_ORIENT_ANGLE: Short\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n

```

```

    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGSolidcolorElement](https://developer.mozilla.org/en/docs/Web/API/SVGSolidcolorElement) to Kotlin\n
*/\npublic external abstract class SVGSolidcolorElement : SVGElement {\n    companion object {\n        val
ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val
CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE:
Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGGradientElement](https://developer.mozilla.org/en/docs/Web/API/SVGGradientElement) to Kotlin\n
*/\npublic external abstract class SVGGradientElement : SVGElement, SVGURIReference, SVGUnitTypes {\n
    open val gradientUnits: SVGAnimatedEnumeration\n    open val gradientTransform: SVGAnimatedTransformList\n
    open val spreadMethod: SVGAnimatedEnumeration\n\n    companion object {\n        val
SVG_SPREADMETHOD_UNKNOWN: Short\n        val SVG_SPREADMETHOD_PAD: Short\n        val
SVG_SPREADMETHOD_REFLECT: Short\n        val SVG_SPREADMETHOD_REPEAT: Short\n        val
SVG_UNIT_TYPE_UNKNOWN: Short\n        val SVG_UNIT_TYPE_USERSPACEONUSE: Short\n        val
SVG_UNIT_TYPE_OBJECTBOUNDINGBOX: Short\n        val ELEMENT_NODE: Short\n        val
ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n        val CDATA_SECTION_NODE: Short\n        val
ENTITY_REFERENCE_NODE: Short\n        val ENTITY_NODE: Short\n        val
PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE: Short\n        val
DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGLinearGradientElement](https://developer.mozilla.org/en/docs/Web/API/SVGLinearGradientElement) to
Kotlin\n */\npublic external abstract class SVGLinearGradientElement : SVGGradientElement {\n    open val x1:
SVGAnimatedLength\n    open val y1: SVGAnimatedLength\n    open val x2: SVGAnimatedLength\n    open val
y2: SVGAnimatedLength\n\n    companion object {\n        val SVG_SPREADMETHOD_UNKNOWN: Short\n
        val SVG_SPREADMETHOD_PAD: Short\n        val SVG_SPREADMETHOD_REFLECT: Short\n        val
SVG_SPREADMETHOD_REPEAT: Short\n        val SVG_UNIT_TYPE_UNKNOWN: Short\n        val
SVG_UNIT_TYPE_USERSPACEONUSE: Short\n        val SVG_UNIT_TYPE_OBJECTBOUNDINGBOX:
Short\n        val ELEMENT_NODE: Short\n        val ATTRIBUTE_NODE: Short\n        val TEXT_NODE: Short\n
        val CDATA_SECTION_NODE: Short\n        val ENTITY_REFERENCE_NODE: Short\n        val
ENTITY_NODE: Short\n        val PROCESSING_INSTRUCTION_NODE: Short\n        val COMMENT_NODE:
Short\n        val DOCUMENT_NODE: Short\n        val DOCUMENT_TYPE_NODE: Short\n        val
DOCUMENT_FRAGMENT_NODE: Short\n        val NOTATION_NODE: Short\n        val
DOCUMENT_POSITION_DISCONNECTED: Short\n        val DOCUMENT_POSITION_PRECEDING: Short\n
        val DOCUMENT_POSITION_FOLLOWING: Short\n        val DOCUMENT_POSITION_CONTAINS: Short\n
        val DOCUMENT_POSITION_CONTAINED_BY: Short\n        val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[SVGRadialGradientElement](https://developer.mozilla.org/en/docs/Web/API/SVGRadialGradientElement) to
Kotlin\n */\npublic external abstract class SVGRadialGradientElement : SVGGradientElement {\n    open val cx:

```

```

SVGAnimatedLength\n  open val cy: SVGAnimatedLength\n  open val r: SVGAnimatedLength\n  open val fx:
SVGAnimatedLength\n  open val fy: SVGAnimatedLength\n  open val fr: SVGAnimatedLength\n\n companion
object {\n    val SVG_SPREADMETHOD_UNKNOWN: Short\n    val SVG_SPREADMETHOD_PAD:
Short\n    val SVG_SPREADMETHOD_REFLECT: Short\n    val SVG_SPREADMETHOD_REPEAT:
Short\n    val SVG_UNIT_TYPE_UNKNOWN: Short\n    val SVG_UNIT_TYPE_USERSPACEONUSE:
Short\n    val SVG_UNIT_TYPE_OBJECTBOUNDINGBOX: Short\n    val ELEMENT_NODE: Short\n
val ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n
val ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n\npublic external abstract class
SVGMeshGradientElement : SVGGradientElement {\n  companion object {\n    val
SVG_SPREADMETHOD_UNKNOWN: Short\n    val SVG_SPREADMETHOD_PAD: Short\n    val
SVG_SPREADMETHOD_REFLECT: Short\n    val SVG_SPREADMETHOD_REPEAT: Short\n    val
SVG_UNIT_TYPE_UNKNOWN: Short\n    val SVG_UNIT_TYPE_USERSPACEONUSE: Short\n    val
SVG_UNIT_TYPE_OBJECTBOUNDINGBOX: Short\n    val ELEMENT_NODE: Short\n    val
ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n\npublic external abstract class
SVGMeshrowElement : SVGElement {\n  companion object {\n    val ELEMENT_NODE: Short\n    val
ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n  }\n\npublic external abstract class
SVGMeshpatchElement : SVGElement {\n  companion object {\n    val ELEMENT_NODE: Short\n    val
ATTRIBUTE_NODE: Short\n    val TEXT_NODE: Short\n    val CDATA_SECTION_NODE: Short\n    val
ENTITY_REFERENCE_NODE: Short\n    val ENTITY_NODE: Short\n    val
PROCESSING_INSTRUCTION_NODE: Short\n    val COMMENT_NODE: Short\n    val
DOCUMENT_NODE: Short\n    val DOCUMENT_TYPE_NODE: Short\n    val
DOCUMENT_FRAGMENT_NODE: Short\n    val NOTATION_NODE: Short\n    val
DOCUMENT_POSITION_DISCONNECTED: Short\n    val DOCUMENT_POSITION_PRECEDING: Short\n
    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val

```



[SVGCursorElement](https://developer.mozilla.org/en/docs/Web/API/SVGCursorElement) to Kotlin\n \* public external abstract class SVGCursorElement : SVGElement, SVGURIReference {\n open val x: SVGAnimatedLength\n open val y: SVGAnimatedLength\n\n companion object {\n val ELEMENT\_NODE: Short\n val ATTRIBUTE\_NODE: Short\n val TEXT\_NODE: Short\n val CDATA\_SECTION\_NODE: Short\n val ENTITY\_REFERENCE\_NODE: Short\n val ENTITY\_NODE: Short\n val PROCESSING\_INSTRUCTION\_NODE: Short\n val COMMENT\_NODE: Short\n val DOCUMENT\_NODE: Short\n val DOCUMENT\_TYPE\_NODE: Short\n val DOCUMENT\_FRAGMENT\_NODE: Short\n val NOTATION\_NODE: Short\n val DOCUMENT\_POSITION\_DISCONNECTED: Short\n val DOCUMENT\_POSITION\_PRECEDING: Short\n val DOCUMENT\_POSITION\_FOLLOWING: Short\n val DOCUMENT\_POSITION\_CONTAINS: Short\n val DOCUMENT\_POSITION\_CONTAINED\_BY: Short\n val DOCUMENT\_POSITION\_IMPLEMENTATION\_SPECIFIC: Short\n }\n}\n\n/\*\*\n \* Exposes the JavaScript [SVGScriptElement](https://developer.mozilla.org/en/docs/Web/API/SVGScriptElement) to Kotlin\n \* public external abstract class SVGScriptElement : SVGElement, SVGURIReference, HTMLOrSVGScriptElement {\n open var type: String\n open var crossOrigin: String?\n\n companion object {\n val ELEMENT\_NODE: Short\n val ATTRIBUTE\_NODE: Short\n val TEXT\_NODE: Short\n val CDATA\_SECTION\_NODE: Short\n val ENTITY\_REFERENCE\_NODE: Short\n val ENTITY\_NODE: Short\n val PROCESSING\_INSTRUCTION\_NODE: Short\n val COMMENT\_NODE: Short\n val DOCUMENT\_NODE: Short\n val DOCUMENT\_TYPE\_NODE: Short\n val DOCUMENT\_FRAGMENT\_NODE: Short\n val NOTATION\_NODE: Short\n val DOCUMENT\_POSITION\_DISCONNECTED: Short\n val DOCUMENT\_POSITION\_PRECEDING: Short\n val DOCUMENT\_POSITION\_FOLLOWING: Short\n val DOCUMENT\_POSITION\_CONTAINS: Short\n val DOCUMENT\_POSITION\_CONTAINED\_BY: Short\n val DOCUMENT\_POSITION\_IMPLEMENTATION\_SPECIFIC: Short\n }\n}\n\n/\*\*\n \* Exposes the JavaScript [SVGAEElement](https://developer.mozilla.org/en/docs/Web/API/SVGAEElement) to Kotlin\n \* public external abstract class SVGAEElement : SVGGraphicsElement, SVGURIReference {\n open val target: SVGAnimatedString\n open val download: SVGAnimatedString\n open val rel: SVGAnimatedString\n open val relList: SVGAnimatedString\n open val hreflang: SVGAnimatedString\n open val type: SVGAnimatedString\n\n companion object {\n val ELEMENT\_NODE: Short\n val ATTRIBUTE\_NODE: Short\n val TEXT\_NODE: Short\n val CDATA\_SECTION\_NODE: Short\n val ENTITY\_REFERENCE\_NODE: Short\n val ENTITY\_NODE: Short\n val PROCESSING\_INSTRUCTION\_NODE: Short\n val COMMENT\_NODE: Short\n val DOCUMENT\_NODE: Short\n val DOCUMENT\_TYPE\_NODE: Short\n val DOCUMENT\_FRAGMENT\_NODE: Short\n val NOTATION\_NODE: Short\n val DOCUMENT\_POSITION\_DISCONNECTED: Short\n val DOCUMENT\_POSITION\_PRECEDING: Short\n val DOCUMENT\_POSITION\_FOLLOWING: Short\n val DOCUMENT\_POSITION\_CONTAINS: Short\n val DOCUMENT\_POSITION\_CONTAINED\_BY: Short\n val DOCUMENT\_POSITION\_IMPLEMENTATION\_SPECIFIC: Short\n }\n}\n\n/\*\*\n \* Exposes the JavaScript [SVGViewElement](https://developer.mozilla.org/en/docs/Web/API/SVGViewElement) to Kotlin\n \* public external abstract class SVGViewElement : SVGElement, SVGFitToViewBox, SVGZoomAndPan {\n\n companion object {\n val SVG\_ZOOMANDPAN\_UNKNOWN: Short\n val SVG\_ZOOMANDPAN\_DISABLE: Short\n val SVG\_ZOOMANDPAN\_MAGNIFY: Short\n val ELEMENT\_NODE: Short\n val ATTRIBUTE\_NODE: Short\n val TEXT\_NODE: Short\n val CDATA\_SECTION\_NODE: Short\n val ENTITY\_REFERENCE\_NODE: Short\n val ENTITY\_NODE: Short\n val PROCESSING\_INSTRUCTION\_NODE: Short\n val COMMENT\_NODE: Short\n val DOCUMENT\_NODE: Short\n val DOCUMENT\_TYPE\_NODE: Short\n val DOCUMENT\_FRAGMENT\_NODE: Short\n val NOTATION\_NODE: Short\n val DOCUMENT\_POSITION\_DISCONNECTED: Short\n val DOCUMENT\_POSITION\_PRECEDING: Short\n

```

    val DOCUMENT_POSITION_FOLLOWING: Short\n    val DOCUMENT_POSITION_CONTAINS: Short\n
    val DOCUMENT_POSITION_CONTAINED_BY: Short\n    val
DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC: Short\n } \n}"/,*\n * Copyright 2010-2021
JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n// NOTE: THIS FILE IS AUTO-
GENERATED, DO NOT EDIT!\n// See github.com/kotlin/dukat for details\n\npackage org.w3c.files\n\nimport
kotlin.js.*\nimport org.khronos.webgl.*\nimport org.w3c.dom.*\nimport org.w3c.dom.events.*\nimport
org.w3c.xhr.*\n\n/**\n * Exposes the JavaScript [Blob](https://developer.mozilla.org/en/docs/Web/API/Blob) to
Kotlin\n */\n\npublic external open class Blob(blobParts: Array<dynamic> = definedExternally, options:
BlobPropertyBag = definedExternally) : MediaPlayer, ImageBitmapSource {\n    open val size: Number\n    open
val type: String\n    open val isClosed: Boolean\n    fun slice(start: Int = definedExternally, end: Int =
definedExternally, contentType: String = definedExternally): Blob\n    fun close()\n}\n\npublic external interface
BlobPropertyBag {\n    var type: String? /* = \"\" */\n    get() = definedExternally\n    set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun BlobPropertyBag(type: String? = \"\"):
BlobPropertyBag {\n    val o = js(\"({})\")\n    o[\"type\"] = type\n    return o\n}\n\n/**\n * Exposes the JavaScript
[File](https://developer.mozilla.org/en/docs/Web/API/File) to Kotlin\n */\n\npublic external open class File(fileBits:
Array<dynamic>, fileName: String, options: FilePropertyBag = definedExternally) : Blob {\n    open val name:
String\n    open val lastModified: Int\n}\n\npublic external interface FilePropertyBag : BlobPropertyBag {\n    var
lastModified: Int?\n    get() = definedExternally\n    set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun FilePropertyBag(lastModified: Int? =
undefined, type: String? = \"\"): FilePropertyBag {\n    val o = js(\"({})\")\n    o[\"lastModified\"] = lastModified\n
o[\"type\"] = type\n    return o\n}\n\n/**\n * Exposes the JavaScript
[FileList](https://developer.mozilla.org/en/docs/Web/API/FileList) to Kotlin\n */\n\npublic external abstract class
FileList : ItemArrayLike<File> {\n    override fun item(index: Int):
File?\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline operator fun FileList.get(index: Int): File?
= asDynamic()[index]\n\n/**\n * Exposes the JavaScript
[FileReader](https://developer.mozilla.org/en/docs/Web/API/FileReader) to Kotlin\n */\n\npublic external open class
FileReader : EventTarget {\n    open val readyState: Short\n    open val result: dynamic\n    open val error:
dynamic\n    var onloadstart: ((ProgressEvent) -> dynamic)?\n    var onprogress: ((ProgressEvent) -> dynamic)?\n
var onload: ((Event) -> dynamic)?\n    var onabort: ((Event) -> dynamic)?\n    var onerror: ((Event) -> dynamic)?\n
var onloadend: ((Event) -> dynamic)?\n    fun readAsArrayBuffer(blob: Blob)\n    fun readAsBinaryString(blob:
Blob)\n    fun readAsText(blob: Blob, label: String = definedExternally)\n    fun readAsDataURL(blob: Blob)\n
fun abort()\n\n    companion object {\n        val EMPTY: Short\n        val LOADING: Short\n        val DONE:
Short\n    }\n}\n\n/**\n * Exposes the JavaScript
[FileReaderSync](https://developer.mozilla.org/en/docs/Web/API/FileReaderSync) to Kotlin\n */\n\npublic external
open class FileReaderSync {\n    fun readAsArrayBuffer(blob: Blob): ArrayBuffer\n    fun readAsBinaryString(blob:
Blob): String\n    fun readAsText(blob: Blob, label: String = definedExternally): String\n    fun
readAsDataURL(blob: Blob): String\n}"/,*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\n// NOTE: THIS FILE IS AUTO-GENERATED, DO NOT EDIT!\n// See
github.com/kotlin/dukat for details\n\npackage org.w3c.notifications\n\nimport kotlin.js.*\nimport
org.khronos.webgl.*\nimport org.w3c.dom.events.*\nimport org.w3c.workers.*\n\n/**\n * Exposes the JavaScript
[Notification](https://developer.mozilla.org/en/docs/Web/API/Notification) to Kotlin\n */\n\npublic external open
class Notification(title: String, options: NotificationOptions = definedExternally) : EventTarget {\n    var onclick:
((MouseEvent) -> dynamic)?\n    var onerror: ((Event) -> dynamic)?\n    open val title: String\n    open val dir:

```

```

NotificationDirection\n open val lang: String\n open val body: String\n open val tag: String\n open val
image: String\n open val icon: String\n open val badge: String\n open val sound: String\n open val vibrate:
Array<out Int>\n open val timestamp: Number\n open val renotify: Boolean\n open val silent: Boolean\n
open val noscreen: Boolean\n open val requireInteraction: Boolean\n open val sticky: Boolean\n open val data:
Any?\n open val actions: Array<out NotificationAction>\n fun close()\n\n companion object {\n val
permission: NotificationPermission\n val maxActions: Int\n fun requestPermission(deprecatedCallback:
(NotificationPermission) -> Unit = definedExternally): Promise<NotificationPermission>\n }\n\n\npublic
external interface NotificationOptions {\n var dir: NotificationDirection? /* = NotificationDirection.AUTO */\n
get() = definedExternally\n set(value) = definedExternally\n var lang: String? /* = "" */\n get() =
definedExternally\n set(value) = definedExternally\n var body: String? /* = "" */\n get() =
definedExternally\n set(value) = definedExternally\n var tag: String? /* = "" */\n get() =
definedExternally\n set(value) = definedExternally\n var image: String?\n get() = definedExternally\n
set(value) = definedExternally\n var icon: String?\n get() = definedExternally\n set(value) =
definedExternally\n var badge: String?\n get() = definedExternally\n set(value) = definedExternally\n
var sound: String?\n get() = definedExternally\n set(value) = definedExternally\n var vibrate: dynamic\n
get() = definedExternally\n set(value) = definedExternally\n var timestamp: Number?\n get() =
definedExternally\n set(value) = definedExternally\n var renotify: Boolean? /* = false */\n get() =
definedExternally\n set(value) = definedExternally\n var silent: Boolean? /* = false */\n get() =
definedExternally\n set(value) = definedExternally\n var noscreen: Boolean? /* = false */\n get() =
definedExternally\n set(value) = definedExternally\n var requireInteraction: Boolean? /* = false */\n get()
= definedExternally\n set(value) = definedExternally\n var sticky: Boolean? /* = false */\n get() =
definedExternally\n set(value) = definedExternally\n var data: Any? /* = null */\n get() =
definedExternally\n set(value) = definedExternally\n var actions: Array<NotificationAction>? /* = arrayOf()
*/\n get() = definedExternally\n set(value) =
definedExternally\n}\n\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun NotificationOptions(dir:
NotificationDirection? = NotificationDirection.AUTO, lang: String? = \"\", body: String? = \"\", tag: String? = \"\",
image: String? = undefined, icon: String? = undefined, badge: String? = undefined, sound: String? = undefined,
vibrate: dynamic = undefined, timestamp: Number? = undefined, renotify: Boolean? = false, silent: Boolean? =
false, noscreen: Boolean? = false, requireInteraction: Boolean? = false, sticky: Boolean? = false, data: Any? = null,
actions: Array<NotificationAction>? = arrayOf()): NotificationOptions {\n val o = js(\"({})\")\n o[\"dir\"] = dir\n
o[\"lang\"] = lang\n o[\"body\"] = body\n o[\"tag\"] = tag\n o[\"image\"] = image\n o[\"icon\"] = icon\n
o[\"badge\"] = badge\n o[\"sound\"] = sound\n o[\"vibrate\"] = vibrate\n o[\"timestamp\"] = timestamp\n
o[\"renotify\"] = renotify\n o[\"silent\"] = silent\n o[\"noscreen\"] = noscreen\n o[\"requireInteraction\"] =
requireInteraction\n o[\"sticky\"] = sticky\n o[\"data\"] = data\n o[\"actions\"] = actions\n return
o\n}\n\n\npublic external interface NotificationAction {\n var action: String?\n var title: String?\n var icon:
String?\n get() = definedExternally\n set(value) =
definedExternally\n}\n\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun NotificationAction(action: String?,
title: String?, icon: String? = undefined): NotificationAction {\n val o = js(\"({})\")\n o[\"action\"] = action\n
o[\"title\"] = title\n o[\"icon\"] = icon\n return o\n}\n\n\npublic external interface GetNotificationOptions {\n var
tag: String? /* = \"\" */\n get() = definedExternally\n set(value) =
definedExternally\n}\n\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun GetNotificationOptions(tag: String? =
\" \"): GetNotificationOptions {\n val o = js(\"({})\")\n o[\"tag\"] = tag\n return o\n}\n\n\n**\n * Exposes the
JavaScript [NotificationEvent](https://developer.mozilla.org/en/docs/Web/API/NotificationEvent) to Kotlin\n
*\n\n\npublic external open class NotificationEvent(type: String, eventInitDict: NotificationEventInit) :
ExtendableEvent {\n open val notification: Notification\n open val action: String\n\n companion object {\n

```

```

val NONE: Short\n    val CAPTURING_PHASE: Short\n    val AT_TARGET: Short\n    val
BUBBLING_PHASE: Short\n    }\n}\n\npublic external interface NotificationEventInit : ExtendableEventInit {\n
var notification: Notification?\n    var action: String? /* = \"\" */\n    get() = definedExternally\n    set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun NotificationEventInit(notification:
Notification?, action: String? = \"\", bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? =
false): NotificationEventInit {\n    val o = js(\"{\}\")\n    o[\"notification\"] = notification\n    o[\"action\"] =
action\n    o[\"bubbles\"] = bubbles\n    o[\"cancelable\"] = cancelable\n    o[\"composed\"] = composed\n    return
o\n}\n\n/* please, don't implement this interface!
*\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\npublic external
interface NotificationPermission {\n    companion object\n}\n\npublic inline val
NotificationPermission.Companion.DEFAULT: NotificationPermission get() =
\"default\".asDynamic().unsafeCast<NotificationPermission>()\n\npublic inline val
NotificationPermission.Companion.DENIED: NotificationPermission get() =
\"denied\".asDynamic().unsafeCast<NotificationPermission>()\n\npublic inline val
NotificationPermission.Companion.GRANTED: NotificationPermission get() =
\"granted\".asDynamic().unsafeCast<NotificationPermission>()\n\n/* please, don't implement this interface!
*\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\npublic external
interface NotificationDirection {\n    companion object\n}\n\npublic inline val
NotificationDirection.Companion.AUTO: NotificationDirection get() =
\"auto\".asDynamic().unsafeCast<NotificationDirection>()\n\npublic inline val
NotificationDirection.Companion.LTR: NotificationDirection get() =
\"ltr\".asDynamic().unsafeCast<NotificationDirection>()\n\npublic inline val
NotificationDirection.Companion.RTL: NotificationDirection get() =
\"rtl\".asDynamic().unsafeCast<NotificationDirection>()\n\n/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin
Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.\n * \n\n// NOTE: THIS FILE IS AUTO-GENERATED, DO NOT EDIT!\n\n//
See github.com/kotlin/dukat for details\n\npackage org.w3c.workers\n\nimport kotlin.js.*\nimport
org.khronos.webgl.*\nimport org.w3c.dom.*\nimport org.w3c.dom.events.*\nimport org.w3c.fetch.*\nimport
org.w3c.notifications.*\n\n/**\n * Exposes the JavaScript
[ServiceWorker](https://developer.mozilla.org/en/docs/Web/API/ServiceWorker) to Kotlin\n *\n\npublic external
abstract class ServiceWorker : EventTarget, AbstractWorker, UnionMessagePortOrServiceWorker,
UnionClientOrMessagePortOrServiceWorker {\n    open val scriptURL: String\n    open val state:
ServiceWorkerState\n    open var onstatechange: ((Event) -> dynamic)?\n    fun postMessage(message: Any?,
transfer: Array<dynamic> = definedExternally)\n}\n\n/**\n * Exposes the JavaScript
[ServiceWorkerRegistration](https://developer.mozilla.org/en/docs/Web/API/ServiceWorkerRegistration) to
Kotlin\n *\n\npublic external abstract class ServiceWorkerRegistration : EventTarget {\n    open val installing:
ServiceWorker?\n    open val waiting: ServiceWorker?\n    open val active: ServiceWorker?\n    open val scope:
String\n    open var onupdatefound: ((Event) -> dynamic)?\n    open val APISpace: dynamic\n    fun update():
Promise<Unit>\n    fun unregister(): Promise<Boolean>\n    fun showNotification(title: String, options:
NotificationOptions = definedExternally): Promise<Unit>\n    fun getNotifications(filter: GetNotificationOptions =
definedExternally): Promise<Array<Notification>>\n    fun methodName(): Promise<dynamic>\n}\n\n/**\n *
Exposes the JavaScript
[ServiceWorkerContainer](https://developer.mozilla.org/en/docs/Web/API/ServiceWorkerContainer) to Kotlin\n
*\n\npublic external abstract class ServiceWorkerContainer : EventTarget {\n    open val controller:
ServiceWorker?\n    open val ready: Promise<ServiceWorkerRegistration>\n    open var oncontrollerchange:
((Event) -> dynamic)?\n    open var onmessage: ((MessageEvent) -> dynamic)?\n    fun register(scriptURL: String,
options: RegistrationOptions = definedExternally): Promise<ServiceWorkerRegistration>\n    fun

```

```

getRegistration(clientURL: String = definedExternally): Promise<Any?>\n fun getRegistrations():
Promise<Array<ServiceWorkerRegistration>>\n fun startMessages()\n\npublic external interface
RegistrationOptions {\n var scope: String?\n get() = definedExternally\n set(value) = definedExternally\n
var type: WorkerType? /* = WorkerType.CLASSIC */\n get() = definedExternally\n set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun RegistrationOptions(scope: String? =
undefined, type: WorkerType? = WorkerType.CLASSIC): RegistrationOptions {\n val o = js(\"({})\")\n
o[\"scope\"] = scope\n o[\"type\"] = type\n return o\n}\n\n/**\n * Exposes the JavaScript
[ServiceWorkerMessageEvent](https://developer.mozilla.org/en/docs/Web/API/ServiceWorkerMessageEvent) to
Kotlin\n */\n\npublic external open class ServiceWorkerMessageEvent(type: String, eventInitDict:
ServiceWorkerMessageEventInit = definedExternally) : Event {\n open val data: Any?\n open val origin:
String\n open val lastEventId: String\n open val source: UnionMessagePortOrServiceWorker?\n open val
ports: Array<out MessagePort>?\n\n companion object {\n val NONE: Short\n val
CAPTURING_PHASE: Short\n val AT_TARGET: Short\n val BUBBLING_PHASE: Short\n
}\n}\n\npublic external interface ServiceWorkerMessageEventInit : EventInit {\n var data: Any?\n get() =
definedExternally\n set(value) = definedExternally\n var origin: String?\n get() = definedExternally\n
set(value) = definedExternally\n var lastEventId: String?\n get() = definedExternally\n set(value) =
definedExternally\n var source: UnionMessagePortOrServiceWorker?\n get() = definedExternally\n
set(value) = definedExternally\n var ports: Array<MessagePort>?\n get() = definedExternally\n set(value)
= definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n\n@kotlin.internal.InlineOnly\n\npublic inline fun ServiceWorkerMessageEventInit(data:
Any? = undefined, origin: String? = undefined, lastEventId: String? = undefined, source:
UnionMessagePortOrServiceWorker? = undefined, ports: Array<MessagePort>? = undefined, bubbles: Boolean? =
false, cancelable: Boolean? = false, composed: Boolean? = false): ServiceWorkerMessageEventInit {\n val o =
js(\"({})\")\n o[\"data\"] = data\n o[\"origin\"] = origin\n o[\"lastEventId\"] = lastEventId\n o[\"source\"] =
source\n o[\"ports\"] = ports\n o[\"bubbles\"] = bubbles\n o[\"cancelable\"] = cancelable\n o[\"composed\"] =
composed\n return o\n}\n\n/**\n * Exposes the JavaScript
[ServiceWorkerGlobalScope](https://developer.mozilla.org/en/docs/Web/API/ServiceWorkerGlobalScope) to
Kotlin\n */\n\npublic external abstract class ServiceWorkerGlobalScope : WorkerGlobalScope {\n open val clients:
Clients\n open val registration: ServiceWorkerRegistration\n open var oninstall: ((Event) -> dynamic)?\n open
var onactivate: ((Event) -> dynamic)?\n open var onfetch: ((FetchEvent) -> dynamic)?\n open var
onforeignfetch: ((Event) -> dynamic)?\n open var onmessage: ((MessageEvent) -> dynamic)?\n open var
onnotificationclick: ((NotificationEvent) -> dynamic)?\n open var onnotificationclose: ((NotificationEvent) ->
dynamic)?\n open var onfunctionalevent: ((Event) -> dynamic)?\n fun skipWaiting():
Promise<Unit>\n}\n\n/**\n * Exposes the JavaScript
[Client](https://developer.mozilla.org/en/docs/Web/API/Client) to Kotlin\n */\n\npublic external abstract class Client :
UnionClientOrMessagePortOrServiceWorker {\n open val url: String\n open val frameType: FrameType\n
open val id: String\n fun postMessage(message: Any?, transfer: Array<dynamic> = definedExternally)\n}\n\n/**\n
* Exposes the JavaScript [WindowClient](https://developer.mozilla.org/en/docs/Web/API/WindowClient) to
Kotlin\n */\n\npublic external abstract class WindowClient : Client {\n open val visibilityState: dynamic\n open
val focused: Boolean\n fun focus(): Promise<WindowClient>\n fun navigate(url: String):
Promise<WindowClient>\n}\n\n/**\n * Exposes the JavaScript
[Clients](https://developer.mozilla.org/en/docs/Web/API/Clients) to Kotlin\n */\n\npublic external abstract class
Clients {\n fun get(id: String): Promise<Any?>\n fun matchAll(options: ClientQueryOptions =
definedExternally): Promise<Array<Client>>\n fun openWindow(url: String): Promise<WindowClient?>\n fun
claim(): Promise<Unit>\n}\n\npublic external interface ClientQueryOptions {\n var includeUncontrolled:
Boolean? /* = false */\n get() = definedExternally\n set(value) = definedExternally\n var type:
ClientType? /* = ClientType.WINDOW */\n get() = definedExternally\n set(value) =

```

```

definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun
ClientQueryOptions(includeUncontrolled: Boolean? = false, type: ClientType? = ClientType.WINDOW):
ClientQueryOptions {\n val o = js(\"({})\")\n o[\"includeUncontrolled\"] = includeUncontrolled\n o[\"type\"] =
type\n return o\n}\n\n/**\n * Exposes the JavaScript
[ExtendableEvent](https://developer.mozilla.org/en/docs/Web/API/ExtendableEvent) to Kotlin\n *\npublic external
open class ExtendableEvent(type: String, eventInitDict: ExtendableEventInit = definedExternally) : Event {\n fun
waitUntil(f: Promise<Any?>)\n\n companion object {\n val NONE: Short\n val CAPTURING_PHASE:
Short\n val AT_TARGET: Short\n val BUBBLING_PHASE: Short\n }\n}\n\npublic external interface
ExtendableEventInit : EventInit\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun ExtendableEventInit(bubbles:
Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false): ExtendableEventInit {\n val o =
js(\"({})\")\n o[\"bubbles\"] = bubbles\n o[\"cancelable\"] = cancelable\n o[\"composed\"] = composed\n
return o\n}\n\n/**\n * Exposes the JavaScript
[InstallEvent](https://developer.mozilla.org/en/docs/Web/API/InstallEvent) to Kotlin\n *\npublic external open
class InstallEvent(type: String, eventInitDict: ExtendableEventInit = definedExternally) : ExtendableEvent {\n fun
registerForeignFetch(options: ForeignFetchOptions)\n\n companion object {\n val NONE: Short\n val
CAPTURING_PHASE: Short\n val AT_TARGET: Short\n val BUBBLING_PHASE: Short\n
}\n}\n\npublic external interface ForeignFetchOptions {\n var scopes: Array<String>?\n var origins:
Array<String>?\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun ForeignFetchOptions(scopes:
Array<String>?, origins: Array<String>?): ForeignFetchOptions {\n val o = js(\"({})\")\n o[\"scopes\"] =
scopes\n o[\"origins\"] = origins\n return o\n}\n\n/**\n * Exposes the JavaScript
[FetchEvent](https://developer.mozilla.org/en/docs/Web/API/FetchEvent) to Kotlin\n *\npublic external open class
FetchEvent(type: String, eventInitDict: FetchEventInit) : ExtendableEvent {\n open val request: Request\n open
val clientId: String?\n open val isReload: Boolean\n fun respondWith(r: Promise<Response>)\n\n companion
object {\n val NONE: Short\n val CAPTURING_PHASE: Short\n val AT_TARGET: Short\n val
BUBBLING_PHASE: Short\n }\n}\n\npublic external interface FetchEventInit : ExtendableEventInit {\n var
request: Request?\n var clientId: String? /* = null */\n get() = definedExternally\n set(value) =
definedExternally\n var isReload: Boolean? /* = false */\n get() = definedExternally\n set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun FetchEventInit(request: Request?,
clientId: String? = null, isReload: Boolean? = false, bubbles: Boolean? = false, cancelable: Boolean? = false,
composed: Boolean? = false): FetchEventInit {\n val o = js(\"({})\")\n o[\"request\"] = request\n o[\"clientId\"]
= clientId\n o[\"isReload\"] = isReload\n o[\"bubbles\"] = bubbles\n o[\"cancelable\"] = cancelable\n
o[\"composed\"] = composed\n return o\n}\n\npublic external open class ForeignFetchEvent(type: String,
eventInitDict: ForeignFetchEventInit) : ExtendableEvent {\n open val request: Request\n open val origin:
String\n fun respondWith(r: Promise<ForeignFetchResponse>)\n\n companion object {\n val NONE:
Short\n val CAPTURING_PHASE: Short\n val AT_TARGET: Short\n val BUBBLING_PHASE:
Short\n }\n}\n\npublic external interface ForeignFetchEventInit : ExtendableEventInit {\n var request:
Request?\n var origin: String? /* = \"null\" */\n get() = definedExternally\n set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun ForeignFetchEventInit(request:
Request?, origin: String? = \"null\", bubbles: Boolean? = false, cancelable: Boolean? = false, composed: Boolean? =
false): ForeignFetchEventInit {\n val o = js(\"({})\")\n o[\"request\"] = request\n o[\"origin\"] = origin\n
o[\"bubbles\"] = bubbles\n o[\"cancelable\"] = cancelable\n o[\"composed\"] = composed\n return
o\n}\n\npublic external interface ForeignFetchResponse {\n var response: Response?\n var origin: String?\n
get() = definedExternally\n set(value) = definedExternally\n var headers: Array<String>?\n get() =

```

```

definedExternally\n    set(value) = definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun ForeignFetchResponse(response:
Response?, origin: String? = undefined, headers: Array<String>? = undefined): ForeignFetchResponse {\n    val o =
js(\"({})\")\n    o[\"response\"] = response\n    o[\"origin\"] = origin\n    o[\"headers\"] = headers\n    return
o\n}\n\n/**\n * Exposes the JavaScript
[ExtendableMessageEvent](https://developer.mozilla.org/en/docs/Web/API/ExtendableMessageEvent) to Kotlin\n
*/\npublic external open class ExtendableMessageEvent(type: String, eventInitDict: ExtendableMessageEventInit =
definedExternally) : ExtendableEvent {\n    open val data: Any?\n    open val origin: String\n    open val lastEventId:
String\n    open val source: UnionClientOrMessagePortOrServiceWorker?\n    open val ports: Array<out
MessagePort>?\n\n    companion object {\n        val NONE: Short\n        val CAPTURING_PHASE: Short\n        val
AT_TARGET: Short\n        val BUBBLING_PHASE: Short\n    }\n}\n\npublic external interface
ExtendableMessageEventInit : ExtendableEventInit {\n    var data: Any?\n        get() = definedExternally\n
set(value) = definedExternally\n    var origin: String?\n        get() = definedExternally\n        set(value) =
definedExternally\n    var lastEventId: String?\n        get() = definedExternally\n        set(value) =
definedExternally\n    var source: UnionClientOrMessagePortOrServiceWorker?\n        get() = definedExternally\n
set(value) = definedExternally\n    var ports: Array<MessagePort>?\n        get() = definedExternally\n
set(value) = definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun ExtendableMessageEventInit(data:
Any? = undefined, origin: String? = undefined, lastEventId: String? = undefined, source:
UnionClientOrMessagePortOrServiceWorker? = undefined, ports: Array<MessagePort>? = undefined, bubbles:
Boolean? = false, cancelable: Boolean? = false, composed: Boolean? = false): ExtendableMessageEventInit {\n
    val o = js(\"({})\")\n    o[\"data\"] = data\n    o[\"origin\"] = origin\n    o[\"lastEventId\"] = lastEventId\n
    o[\"source\"] = source\n    o[\"ports\"] = ports\n    o[\"bubbles\"] = bubbles\n    o[\"cancelable\"] = cancelable\n
    o[\"composed\"] = composed\n    return o\n}\n\n/**\n * Exposes the JavaScript
[Cache](https://developer.mozilla.org/en/docs/Web/API/Cache) to Kotlin\n
*/\npublic external abstract class Cache
{\n    fun match(request: dynamic, options: CacheQueryOptions = definedExternally): Promise<Any?>\n    fun
matchAll(request: dynamic = definedExternally, options: CacheQueryOptions = definedExternally):
Promise<Array<Response>>\n    fun add(request: dynamic): Promise<Unit>\n    fun addAll(requests:
Array<dynamic>): Promise<Unit>\n    fun put(request: dynamic, response: Response): Promise<Unit>\n    fun
delete(request: dynamic, options: CacheQueryOptions = definedExternally): Promise<Boolean>\n    fun
keys(request: dynamic = definedExternally, options: CacheQueryOptions = definedExternally):
Promise<Array<Request>>\n}\n\npublic external interface CacheQueryOptions {\n    var ignoreSearch: Boolean? /*
= false */\n        get() = definedExternally\n        set(value) = definedExternally\n    var ignoreMethod: Boolean? /*
= false */\n        get() = definedExternally\n        set(value) = definedExternally\n    var ignoreVary: Boolean? /*
= false */\n        get() = definedExternally\n        set(value) = definedExternally\n    var cacheName: String?\n
get() = definedExternally\n        set(value) = definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun CacheQueryOptions(ignoreSearch:
Boolean? = false, ignoreMethod: Boolean? = false, ignoreVary: Boolean? = false, cacheName: String? = undefined):
CacheQueryOptions {\n    val o = js(\"({})\")\n    o[\"ignoreSearch\"] = ignoreSearch\n    o[\"ignoreMethod\"] =
ignoreMethod\n    o[\"ignoreVary\"] = ignoreVary\n    o[\"cacheName\"] = cacheName\n    return o\n}\n\npublic
external interface CacheBatchOperation {\n    var type: String?\n        get() = definedExternally\n        set(value) =
definedExternally\n    var request: Request?\n        get() = definedExternally\n        set(value) = definedExternally\n
var response: Response?\n        get() = definedExternally\n        set(value) = definedExternally\n    var options:
CacheQueryOptions?\n        get() = definedExternally\n        set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun CacheBatchOperation(type: String? =
undefined, request: Request? = undefined, response: Response? = undefined, options: CacheQueryOptions? =
undefined): CacheBatchOperation {\n    val o = js(\"({})\")\n    o[\"type\"] = type\n    o[\"request\"] = request\n

```

```

o["response"] = response\n  o["options"] = options\n  return o\n}\n\n/**\n * Exposes the JavaScript
[CacheStorage](https://developer.mozilla.org/en/docs/Web/API/CacheStorage) to Kotlin\n */\npublic external
abstract class CacheStorage {\n  fun match(request: dynamic, options: CacheQueryOptions = definedExternally):
Promise<Any?>\n  fun has(cacheName: String): Promise<Boolean>\n  fun open(cacheName: String):
Promise<Cache>\n  fun delete(cacheName: String): Promise<Boolean>\n  fun keys():
Promise<Array<String>>\n}\n\npublic external open class FunctionalEvent : ExtendableEvent {\n  companion
object {\n    val NONE: Short\n    val CAPTURING_PHASE: Short\n    val AT_TARGET: Short\n    val
BUBBLING_PHASE: Short\n  }\n}\n\npublic external interface UnionMessagePortOrServiceWorker\n\npublic
external interface UnionClientOrMessagePortOrServiceWorker\n\n/* please, don't implement this interface!
*\n */\n@JsName("null")\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\n\npublic external
interface ServiceWorkerState {\n  companion object\n}\n\npublic inline val
ServiceWorkerState.Companion.INSTALLING: ServiceWorkerState get() =
"installing".asDynamic().unsafeCast<ServiceWorkerState>()\n\npublic inline val
ServiceWorkerState.Companion.INSTALLED: ServiceWorkerState get() =
"installed".asDynamic().unsafeCast<ServiceWorkerState>()\n\npublic inline val
ServiceWorkerState.Companion.ACTIVATING: ServiceWorkerState get() =
"activating".asDynamic().unsafeCast<ServiceWorkerState>()\n\npublic inline val
ServiceWorkerState.Companion.ACTIVATED: ServiceWorkerState get() =
"activated".asDynamic().unsafeCast<ServiceWorkerState>()\n\npublic inline val
ServiceWorkerState.Companion.REDUNDANT: ServiceWorkerState get() =
"redundant".asDynamic().unsafeCast<ServiceWorkerState>()\n\n/* please, don't implement this interface!
*\n */\n@JsName("null")\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\n\npublic external
interface FrameType {\n  companion object\n}\n\npublic inline val FrameType.Companion.AUXILIARY:
FrameType get() = "auxiliary".asDynamic().unsafeCast<FrameType>()\n\npublic inline val
FrameType.Companion.TOP_LEVEL: FrameType get() = "top-
level".asDynamic().unsafeCast<FrameType>()\n\npublic inline val FrameType.Companion.NESTED: FrameType
get() = "nested".asDynamic().unsafeCast<FrameType>()\n\npublic inline val FrameType.Companion.NONE:
FrameType get() = "none".asDynamic().unsafeCast<FrameType>()\n\n/* please, don't implement this interface!
*\n */\n@JsName("null")\n@Suppress("NESTED_CLASS_IN_EXTERNAL_INTERFACE")\n\npublic external
interface ClientType {\n  companion object\n}\n\npublic inline val ClientType.Companion.WINDOW: ClientType
get() = "window".asDynamic().unsafeCast<ClientType>()\n\npublic inline val ClientType.Companion.WORKER:
ClientType get() = "worker".asDynamic().unsafeCast<ClientType>()\n\npublic inline val
ClientType.Companion.SHAREDWORKER: ClientType get() =
"sharedworker".asDynamic().unsafeCast<ClientType>()\n\npublic inline val ClientType.Companion.ALL:
ClientType get() = "all".asDynamic().unsafeCast<ClientType>()"/**\n * Copyright 2010-2021 JetBrains s.r.o. and
Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n */\n\n// NOTE: THIS FILE IS AUTO-GENERATED, DO NOT
EDIT!\n\n// See github.com/kotlin/dukat for details\n\npackage org.w3c.xhr\n\nimport kotlin.js.*\nimport
org.khronos.webgl.*\nimport org.w3c.dom.*\nimport org.w3c.dom.events.*\nimport org.w3c.files.*\n\n/**\n *
Exposes the JavaScript
[XMLHttpRequestEventTarget](https://developer.mozilla.org/en/docs/Web/API/XMLHttpRequestEventTarget) to
Kotlin\n */\n\npublic external abstract class XMLHttpRequestEventTarget : EventTarget {\n  open var onloadstart:
((ProgressEvent) -> dynamic)?\n  open var onprogress: ((ProgressEvent) -> dynamic)?\n  open var onabort:
((Event) -> dynamic)?\n  open var onerror: ((Event) -> dynamic)?\n  open var onload: ((Event) -> dynamic)?\n
open var ontimeout: ((Event) -> dynamic)?\n  open var onloadend: ((Event) -> dynamic)?\n}\n\npublic external
abstract class XMLHttpRequestUpload : XMLHttpRequestEventTarget\n\n/**\n * Exposes the JavaScript
[XMLHttpRequest](https://developer.mozilla.org/en/docs/Web/API/XMLHttpRequest) to Kotlin\n */\n\npublic
external open class XMLHttpRequest : XMLHttpRequestEventTarget {\n  var onreadystatechange: ((Event) ->

```

```

dynamic)?\n open val readyState: Short\n var timeout: Int\n var withCredentials: Boolean\n open val upload:
XMLHttpRequestUpload\n open val responseURL: String\n open val status: Short\n open val statusText:
String\n var responseType: XMLHttpRequestResponseType\n open val response: Any?\n open val
responseText: String\n open val responseXML: Document?\n fun open(method: String, url: String)\n fun
open(method: String, url: String, async: Boolean, username: String? = definedExternally, password: String? =
definedExternally)\n fun setRequestHeader(name: String, value: String)\n fun send(body: dynamic =
definedExternally)\n fun abort()\n fun getResponseHeader(name: String): String?\n fun
getAllResponseHeaders(): String\n fun overrideMimeType(mime: String)\n\n companion object {\n val
UNSENT: Short\n val OPENED: Short\n val HEADERS_RECEIVED: Short\n val LOADING:
Short\n val DONE: Short\n }\n}\n\n/**\n * Exposes the JavaScript
[FormData](https://developer.mozilla.org/en/docs/Web/API/FormData) to Kotlin\n */\npublic external open class
FormData(form: HTMLFormElement = definedExternally) {\n fun append(name: String, value: String)\n fun
append(name: String, value: Blob, filename: String = definedExternally)\n fun delete(name: String)\n fun
get(name: String): dynamic\n fun getAll(name: String): Array<dynamic>\n fun has(name: String): Boolean\n
fun set(name: String, value: String)\n fun set(name: String, value: Blob, filename: String =
definedExternally)\n}\n\n/**\n * Exposes the JavaScript
[ProgressEvent](https://developer.mozilla.org/en/docs/Web/API/ProgressEvent) to Kotlin\n */\npublic external open
class ProgressEvent(type: String, eventInitDict: ProgressEventInit = definedExternally) : Event {\n open val
lengthComputable: Boolean\n open val loaded: Number\n open val total: Number\n\n companion object {\n
val NONE: Short\n val CAPTURING_PHASE: Short\n val AT_TARGET: Short\n val
BUBBLING_PHASE: Short\n }\n}\n\npublic external interface ProgressEventInit : EventInit {\n var
lengthComputable: Boolean? /* = false */\n get() = definedExternally\n set(value) = definedExternally\n
var loaded: Number? /* = 0 */\n get() = definedExternally\n set(value) = definedExternally\n var total:
Number? /* = 0 */\n get() = definedExternally\n set(value) =
definedExternally\n}\n\n@Suppress(\"INVISIBLE_REFERENCE\",
\"INVISIBLE_MEMBER\")\n@kotlin.internal.InlineOnly\npublic inline fun ProgressEventInit(lengthComputable:
Boolean? = false, loaded: Number? = 0, total: Number? = 0, bubbles: Boolean? = false, cancelable: Boolean? =
false, composed: Boolean? = false): ProgressEventInit {\n val o = js(\"{\}\")\n o[\"lengthComputable\"] =
lengthComputable\n o[\"loaded\"] = loaded\n o[\"total\"] = total\n o[\"bubbles\"] = bubbles\n
o[\"cancelable\"] = cancelable\n o[\"composed\"] = composed\n return o\n}\n\n/* please, don't implement this
interface! */\n@JsName(\"null\")\n@Suppress(\"NESTED_CLASS_IN_EXTERNAL_INTERFACE\")\npublic
external interface XMLHttpRequestResponseType {\n companion object\n}\n\npublic inline val
XMLHttpRequestResponseType.Companion.EMPTY: XMLHttpRequestResponseType get() =
\"\".asDynamic().unsafeCast<XMLHttpRequestResponseType>()\n\npublic inline val
XMLHttpRequestResponseType.Companion.ARRAYBUFFER: XMLHttpRequestResponseType get() =
\"arraybuffer\".asDynamic().unsafeCast<XMLHttpRequestResponseType>()\n\npublic inline val
XMLHttpRequestResponseType.Companion.BLOB: XMLHttpRequestResponseType get() =
\"blob\".asDynamic().unsafeCast<XMLHttpRequestResponseType>()\n\npublic inline val
XMLHttpRequestResponseType.Companion.DOCUMENT: XMLHttpRequestResponseType get() =
\"document\".asDynamic().unsafeCast<XMLHttpRequestResponseType>()\n\npublic inline val
XMLHttpRequestResponseType.Companion.JSON: XMLHttpRequestResponseType get() =
\"json\".asDynamic().unsafeCast<XMLHttpRequestResponseType>()\n\npublic inline val
XMLHttpRequestResponseType.Companion.TEXT: XMLHttpRequestResponseType get() =
\"text\".asDynamic().unsafeCast<XMLHttpRequestResponseType>()\n\n}/*\n * Copyright 2010-2018 JetBrains s.r.o.
and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license
that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin\n\nimport
kotlin.annotation.AnnotationRetention.BINARY\nimport kotlin.annotation.AnnotationRetention.SOURCE\nimport
kotlin.annotation.AnnotationTarget.*\nimport kotlin.internal.RequireKotlin\nimport

```

```

kotlin.internal.RequireKotlinVersionKind\nimport kotlin.reflect.KClass\n\n/**\n * Signals that the annotated
annotation class is a marker of an experimental API.\n *\n * Any declaration annotated with that marker is
considered an experimental declaration\n * and its call sites should accept the experimental aspect of it either by
using [UseExperimental],\n * or by being annotated with that marker themselves, effectively causing further
propagation of that experimental aspect.\n *\n * This class is deprecated in favor of a more general approach
provided by [RequiresOptIn]/[OptIn].\n
*\n@Target(ANNOTATION_CLASS)\n@Retention(BINARY)\n@SinceKotlin("1.2")\n@RequireKotlin("1.2.50
\", versionKind = RequireKotlinVersionKind.COMPILER_VERSION)\n@Deprecated(\"Please use RequiresOptIn
instead.\")\npublic annotation class Experimental(val level: Level = Level.ERROR) {\n    /**\n     * Severity of the
diagnostic that should be reported on usages of experimental API which did not explicitly accept the experimental
aspect\n     * of that API either by using [UseExperimental] or by being annotated with the corresponding marker
annotation.\n     *\n     * public enum class Level {\n         /** Specifies that a warning should be reported on incorrect
usages of this experimental API. *\n         WARNING,\n         /** Specifies that an error should be reported on
incorrect usages of this experimental API. *\n         ERROR,\n     }\n}\n\n/**\n * Allows to use experimental API
denoted by the given markers in the annotated file, declaration, or expression.\n * If a declaration is annotated with
[UseExperimental], its usages are **not** required to opt-in to that experimental API.\n *\n * This class is
deprecated in favor of a more general approach provided by [RequiresOptIn]/[OptIn].\n *\n@Target(\n    CLASS,
PROPERTY, LOCAL_VARIABLE, VALUE_PARAMETER, CONSTRUCTOR, FUNCTION,
PROPERTY_GETTER, PROPERTY_SETTER, EXPRESSION, FILE,
TYPEALIAS)\n)\n@Retention(SOURCE)\n@SinceKotlin("1.2")\n@RequireKotlin("1.2.50\", versionKind =
RequireKotlinVersionKind.COMPILER_VERSION)\n@Deprecated(\"Please use OptIn instead.\",
ReplaceWith(\"OptIn(*markerClass)\", \"kotlin.OptIn\"))\npublic annotation class UseExperimental(\n    vararg val
markerClass: KClass<out Annotation>)\n)\n\n@Target(CLASS, PROPERTY, CONSTRUCTOR, FUNCTION,
TYPEALIAS)\n@Retention(BINARY)\ninternal annotation class WasExperimental(\n    vararg val markerClass:
KClass<out Annotation>)\n)\n\", \"package kotlin\n\nimport kotlin.annotation.AnnotationTarget.*\n\n/**\n * This
annotation marks the standard library API that is considered experimental and is not subject to the\n * [general
compatibility guarantees](https://kotlinlang.org/docs/reference/evolution/components-stability.html) given for the
standard library:\n * the behavior of such API may be changed or the API may be removed completely in any
further release.\n *\n * > Beware using the annotated API especially if you're developing a library, since your library
might become binary incompatible\n * with the future versions of the standard library.\n *\n * Any usage of a
declaration annotated with `@ExperimentalStdlibApi` must be accepted either by\n * annotating that usage with the
[OptIn] annotation, e.g. `@OptIn(ExperimentalStdlibApi::class)`,\n * or by using the compiler argument `-Xopt-
in=kotlin.ExperimentalStdlibApi`. \n *\n@Suppress(\"DEPRECATION\")\n@Experimental(level =
Experimental.Level.ERROR)\n@RequiresOptIn(level =
RequiresOptIn.Level.ERROR)\n@Retention(AnnotationRetention.BINARY)\n@Target(\n    CLASS,\n
ANNOTATION_CLASS,\n    PROPERTY,\n    FIELD,\n    LOCAL_VARIABLE,\n    VALUE_PARAMETER,\n
CONSTRUCTOR,\n    FUNCTION,\n    PROPERTY_GETTER,\n    PROPERTY_SETTER,\n
TYPEALIAS)\n)\n@MustBeDocumented\n@SinceKotlin("1.3")\npublic annotation class
ExperimentalStdlibApi\n\", \"/*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n *\npackage kotlin\n\nimport kotlin.annotation.AnnotationTarget.*\nimport
kotlin.experimental.ExperimentalTypeInference\n\n/**\n * Allows to infer generic type arguments of a function
from the calls in the annotated function parameter of that function.\n *\n * When this annotation is placed on a
generic function parameter of a function,\n * it enables to infer the type arguments of that generic function from the
lambda body passed to that parameter.\n *\n * The calls that affect inference are either members of the receiver type
of an annotated function parameter or\n * extensions for that type. The extensions must be themselves annotated
with `@BuilderInference`. \n *\n * Example: we declare\n * ``\n * fun <T> sequence(@BuilderInference block:
suspend SequenceScope<T>().() -> Unit): Sequence<T>\n * ``\n * and use it like\n * ``\n * val result = sequence {

```

yield("result") } }  
 Here the type argument of the resulting sequence is inferred to `String` from the argument of the [SequenceScope.yield] function, that is called inside the lambda passed to [sequence].  
 Note: this annotation is experimental, see [ExperimentalTypeInference] on how to opt-in for it.

```

@Target(VALUE_PARAMETER, FUNCTION,
PROPERTY)
@Retention(AnnotationRetention.BINARY)
@SinceKotlin("1.3")
@ExperimentalTypeInference
public annotation class BuilderInference
  * Enables overload selection based on the type of the value returned from lambda argument.
  * When two or more function overloads have otherwise the same parameter lists that differ only in the return type of a functional parameter, this annotation enables overload selection by the type of the value returned from the lambda function passed to this functional parameter.
  * Example:
  * @OverloadResolutionByLambdaReturnType
  * fun create(intProducer: () -> Int): Int
  * fun create(doubleProducer: () -> Double): Double
  * val newValue = create { 3.14 }
  * The annotation being applied to one of overloads allows to resolve this ambiguity by analyzing what value is returned from the lambda function.
  * This annotation is also used to discriminate the annotated overloads in case if overload selection still cannot choose one of them even taking in account the result of lambda parameter analysis. In that case a warning is reported.
  * Note: this annotation is experimental, see [ExperimentalTypeInference] on how to opt-in for it.

```

```

@Target(FUNCTION)
@Retention(AnnotationRetention.BINARY)
@SinceKotlin("1.4")
@ExperimentalTypeInference
public annotation class OverloadResolutionByLambdaReturnType
  * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.
  * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.
  * package kotlin
  * import kotlin.annotation.AnnotationTarget
  * import kotlin.internal.RequireKotlin
  * import kotlin.internal.RequireKotlinVersionKind
  * The experimental multiplatform support API marker.
  * Any usage of a declaration annotated with `@ExperimentalMultiplatform` must be accepted either by annotating that usage with the [OptIn] annotation, e.g. `@OptIn(ExperimentalMultiplatform::class)`, or by using the compiler argument `-Xopt-in=kotlin.ExperimentalMultiplatform`.

```

```

@Suppress("DEPRECATION")
@Experimental
@RequiresOptIn
@MustBeDocumented
@Target(CLASS, ANNOTATION_CLASS, PROPERTY, FIELD, LOCAL_VARIABLE, VALUE_PARAMETER, CONSTRUCTOR, FUNCTION, PROPERTY_GETTER, PROPERTY_SETTER, TYPEALIAS)
@Retention(AnnotationRetention.BINARY)
@RequireKotlin("1.2.50", versionKind = RequireKotlinVersionKind.COMPILER_VERSION)
public annotation class ExperimentalMultiplatform
  * Marks an expected annotation class that it isn't required to have actual counterparts in all platforms.
  * This annotation is only applicable to `expect` annotation classes in multi-platform projects and marks that class as `optional`.
  * Optional expected class is allowed to have no corresponding actual class on the platform. Optional annotations can only be used to annotate something, not as types in signatures. If an optional annotation has no corresponding actual class on a platform, the annotation entries where it's used are simply erased when compiling code on that platform.
  * Note: this annotation is experimental, see [ExperimentalMultiplatform] on how to opt-in for it.

```

```

@Target(ANNOTATION_CLASS)
@Retention(AnnotationRetention.BINARY)
@ExperimentalMultiplatform
@RequireKotlin("1.2.50", versionKind = RequireKotlinVersionKind.COMPILER_VERSION)
public annotation class OptionalExpectation
  * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.
  * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.
  * package kotlin
  * import kotlin.annotation.AnnotationRetention.BINARY
  * import kotlin.annotation.AnnotationRetention.SOURCE
  * import kotlin.annotation.AnnotationTarget
  * import kotlin.internal.RequireKotlin
  * import kotlin.reflect.KClass
  * Signals that the annotated annotation class is a marker of an API that requires an explicit opt-in.
  * Call sites of any declaration annotated with that marker should opt in to the API either by using [OptIn], or by being annotated with that marker

```

themselves, effectively causing further propagation of the opt-in requirement.\n \* This class requires opt-in itself and can only be used with the compiler argument `-Xopt-in=kotlin.RequiresOptIn`.\n \* @property message message to be reported on usages of API without an explicit opt-in, or empty string for the default message.\n \*

The default message is: `"This declaration is experimental and its usage should be marked with 'Marker' or '@OptIn(Marker::class)'"`, where `Marker` is the opt-in requirement marker.\n \* @property level specifies how usages of API without an explicit opt-in are reported in code.\n

```
*\n@Target(ANNOTATION_CLASS)\n@Retention(BINARY)\n@SinceKotlin("1.3")\n@RequireKotlin("1.3.70", versionKind = RequireKotlinVersionKind.COMPILER_VERSION)\npublic annotation class RequiresOptIn(\n    val message: String = "",\n    val level: Level = Level.ERROR) {\n    /**\n     * Severity of the diagnostic that should be reported on usages which did not explicitly opt into the API either by using [OptIn] or by being annotated with the corresponding marker annotation.\n     */\n    public enum class Level {\n        /** Specifies that a warning should be reported on incorrect usages of this API. */\n        WARNING,\n        /** Specifies that an error should be reported on incorrect usages of this API. */\n        ERROR,\n    }\n}\n/**\n * Allows to use the API denoted by the given markers in the annotated file, declaration, or expression.\n * If a declaration is annotated with [OptIn], its usages are not required to opt in to that API.\n * This class requires opt-in itself and can only be used with the compiler argument -Xopt-in=kotlin.RequiresOptIn. \n *\n @Target(CLASS, PROPERTY, LOCAL_VARIABLE, VALUE_PARAMETER, CONSTRUCTOR, FUNCTION, PROPERTY_GETTER, PROPERTY_SETTER, EXPRESSION, FILE,
```

```
TYPEALIAS)\n@Retention(SOURCE)\n@SinceKotlin("1.3")\n@RequireKotlin("1.3.70", versionKind = RequireKotlinVersionKind.COMPILER_VERSION)\npublic annotation class OptIn(\n    vararg val markerClass: KClass<out Annotation>)\n"/*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\npackage kotlin.collections\n\nimport kotlin.js.JsName\n\n/**\n * Provides a skeletal implementation of the read-only [Collection] interface.\n */\n * @param E the type of elements contained in the collection. The collection is covariant in its element type.\n *\n @SinceKotlin("1.1")\npublic abstract class AbstractCollection<out E> protected constructor() : Collection<E> {\n    abstract override val size: Int\n    abstract override fun iterator(): Iterator<E>\n    override fun contains(element: @UnsafeVariance E): Boolean = any { it == element }\n    override fun containsAll(elements: Collection<@UnsafeVariance E>): Boolean =\n        elements.all { contains(it) } // use when js will support bound refs: elements.all(this::contains)\n    override fun isEmpty(): Boolean = size == 0\n    override fun toString(): String = joinToString(", ", "[", "]")\n    if (it == this) "(this Collection)" else it.toString()\n }\n /**\n  * Returns new array of type Array<Any?> with the elements of this collection.\n  *\n  @JsName("toArray")\n  protected open fun toArray(): Array<Any?> = copyToArrayImpl(this)\n  /**\n   * Fills the provided [array] or creates new array of the same type and fills it with the elements of this collection.\n   *\n   protected open fun <T> toArray(array: Array<T>): Array<T> = copyToArrayImpl(this, array)\n }\n"/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\npackage kotlin.collections\n\nprivate enum class State {\n    Ready,\n    NotReady,\n    Done,\n    Failed\n}\n/**\n * A base class to simplify implementing iterators so that implementations only have to implement [computeNext] to implement the iterator, calling [done] when the iteration is complete.\n *\n public abstract class AbstractIterator<T> : Iterator<T> {\n    private var state = State.NotReady\n    private var nextValue: T? = null\n    override fun hasNext(): Boolean {\n        require(state != State.Failed)\n        return when (state) {\n            State.Done -> false\n            State.Ready -> true\n            else -> tryToComputeNext()\n        }\n    }\n    override fun next(): T {\n        if (!hasNext()) throw NoSuchElementException()\n        state = State.NotReady\n        @Suppress("UNCHECKED_CAST")\n        return nextValue as T\n    }\n    private fun tryToComputeNext(): Boolean {\n        state = State.Failed\n        computeNext()\n        return state == State.Ready\n    }\n /**\n  * Computes the next item in the iterator.\n  *\n  * This callback method should call one of these two methods:\n  * * [setNext] with the next value of the iteration\n  * * [done] to indicate there are no more elements\n  * * Failure to call either method will
```

```

result in the iteration terminating with a failed state\n    */\n    abstract protected fun computeNext(): Unit\n    /**\n     * Sets the next value in the iteration, called from the [computeNext] function\n     */\n    protected fun\n    setNext(value: T): Unit {\n        nextValue = value\n        state = State.Ready\n    }\n    /**\n     * Sets the state to\n    done so that the iteration terminates.\n     */\n    protected fun done() {\n        state = State.Done\n    }\n}\n\n"/*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code\n    is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n/*\n * Based on\n    GWT AbstractList\n * Copyright 2007 Google Inc.\n */\n\npackage kotlin.collections\n\n/**\n * Provides a skeletal\n    implementation of the read-only [List] interface.\n */\n * This class is intended to help implementing read-only lists\n    so it doesn't support concurrent modification tracking.\n */\n * @param E the type of elements contained in the list.\n    The list is covariant in its element type.\n */\n\n@SinceKotlin("1.1")\npublic abstract class AbstractList<out E>\nprotected constructor() : AbstractCollection<E>(), List<E> {\n    abstract override val size: Int\n    abstract override\n    fun get(index: Int): E\n    override fun iterator(): Iterator<E> = IteratorImpl()\n    override fun indexOf(element:\n    @UnsafeVariance E): Int = indexOfFirst { it == element }\n    override fun lastIndexOf(element:\n    @UnsafeVariance E): Int = indexOfLast { it == element }\n    override fun listIterator(): ListIterator<E> =\n    ListIteratorImpl(0)\n    override fun listIterator(index: Int): ListIterator<E> = ListIteratorImpl(index)\n    override fun subList(fromIndex: Int, toIndex: Int): List<E> = SubList(this, fromIndex, toIndex)\n    private class\n    SubList<out E>(private val list: AbstractList<E>, private val fromIndex: Int, toIndex: Int) : AbstractList<E>(),\n    RandomAccess {\n        private var _size: Int = 0\n        init {\n            checkRangeIndexes(fromIndex, toIndex,\n                list.size)\n            this._size = toIndex - fromIndex\n        }\n        override fun get(index: Int): E {\n            checkElementIndex(index, _size)\n            return list[fromIndex + index]\n        }\n        override val size: Int\n        get() = _size\n    }\n    /**\n     * Compares this list with other list instance with the ordered structural equality.\n     */\n    * @return true, if [other] instance is a [List] of the same size, which contains the same elements in the same\n    order.\n     */\n    override fun equals(other: Any?): Boolean {\n        if (other === this) return true\n        if (other !is\n        List<*>) return false\n        return orderedEquals(this, other)\n    }\n    /**\n     * Returns the hash code value for\n    this list.\n     */\n    override fun hashCode(): Int = orderedHashCode(this)\n    private open inner class IteratorImpl\n    : Iterator<E> {\n        /** the index of the item that will be returned on the next call to [next] ()\n        */\n        protected\n        var index = 0\n        override fun hasNext(): Boolean = index < size\n        override fun next(): E {\n            if\n            (!hasNext()) throw NoSuchElementException()\n            return get(index++)\n        }\n    }\n    /**\n     *\n    Implementation of [ListIterator] for abstract lists.\n     */\n    private open inner class ListIteratorImpl(index: Int) :\n    IteratorImpl(), ListIterator<E> {\n        init {\n            checkPositionIndex(index, this@AbstractList.size)\n            this.index = index\n        }\n        override fun hasNext(): Boolean = index > 0\n        override fun\n        nextIndex(): Int = index\n        override fun previous(): E {\n            if (!hasPrevious()) throw\n            NoSuchElementException()\n            return get(--index)\n        }\n        override fun previousIndex(): Int = index -\n        1\n    }\n    internal companion object {\n        internal fun checkElementIndex(index: Int, size: Int) {\n            if\n            (index < 0 || index >= size) {\n                throw IndexOutOfBoundsException("index: $index, size: $size")\n            }\n        }\n        internal fun checkPositionIndex(index: Int, size: Int) {\n            if (index < 0 || index > size) {\n                throw IndexOutOfBoundsException("index: $index, size: $size")\n            }\n        }\n        internal fun\n        checkRangeIndexes(fromIndex: Int, toIndex: Int, size: Int) {\n            if (fromIndex < 0 || toIndex > size) {\n                throw IndexOutOfBoundsException("fromIndex: $fromIndex, toIndex: $toIndex, size: $size")\n            }\n            if (fromIndex > toIndex) {\n                throw IllegalArgumentException("fromIndex: $fromIndex > toIndex:\n                $toIndex")\n            }\n        }\n        internal fun checkBoundsIndexes(startIndex: Int, endIndex: Int, size: Int) {\n            if (startIndex < 0 || endIndex > size) {\n                throw IndexOutOfBoundsException("startIndex:\n                $startIndex, endIndex: $endIndex, size: $size")\n            }\n            if (startIndex > endIndex) {\n                throw\n                IllegalArgumentException("startIndex: $startIndex > endIndex: $endIndex")\n            }\n        }\n        internal\n        fun orderedHashCode(c: Collection<*>): Int {\n            var hashCode = 1\n            for (e in c) {\n                hashCode = 31 * hashCode + (e?.hashCode() ?: 0)\n            }\n            return hashCode\n        }\n        internal fun\n        orderedEquals(c: Collection<*>, other: Collection<*>): Boolean {\n            if (c.size != other.size) return false\n            val otherIterator = other.iterator()\n            for (elem in c) {\n                val elemOther = otherIterator.next()\n            }\n        }\n    }\n}

```

```

        if (elem != elemOther) {
            return false
        }
        return true
    }
}
"/>
 * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.
 * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.
 *
 * Based on GWT AbstractMap
 * Copyright 2007 Google Inc.
 *
 * Provides a skeletal implementation of the read-only [Map] interface.
 * The implementor is required to implement [entries] property, which should return read-only set of map entries.
 * @param K the type of map keys. The map is invariant in its key type.
 * @param V the type of map values. The map is covariant in its value type.
 *
 * Since Kotlin 1.1
 * public abstract class AbstractMap<K, out V> protected constructor() : Map<K, V> {
 *
 *     override fun containsKey(key: K): Boolean {
 *         return implFindEntry(key) != null
 *     }
 *     override fun containsValue(value: @UnsafeVariance V): Boolean = entries.any { it.value == value }
 *     internal fun containsEntry(entry: Map.Entry<*, *>): Boolean {
 *         // since entry comes from @UnsafeVariance parameters it can be virtually anything
 *         if (entry !is Map.Entry<*, *>) return false
 *         val key = entry.key
 *         val value = entry.value
 *         val ourValue = get(key)
 *         if (value != ourValue) {
 *             return false
 *         }
 *         // Perhaps it was null and we don't contain the key?
 *         if (ourValue == null && !containsKey(key)) {
 *             return false
 *         }
 *         return true
 *     }
 *     /**
 *      * Compares this map with other instance with the ordered structural equality.
 *      * @return true, if [other] instance is a [Map] of the same size, all entries of which are contained in the [entries] set of this map.
 *      */
 *     override fun equals(other: Any?): Boolean {
 *         if (other === this) return true
 *         if (other !is Map<*, *>) return false
 *         if (size != other.size) return false
 *         return other.entries.all { containsEntry(it) }
 *     }
 *     override operator fun get(key: K): V? = implFindEntry(key)?.value
 *     /**
 *      * Returns the hash code value for this map.
 *      * It is the same as the hashCode of [entries] set.
 *      */
 *     override fun hashCode(): Int = entries.hashCode()
 *     override fun isEmpty(): Boolean = size == 0
 *     override val size: Int get() = entries.size
 *     /**
 *      * Returns a read-only [Set] of all keys in this map.
 *      * Accessing this property first time creates a keys view from [entries].
 *      * All subsequent accesses just return the created instance.
 *      */
 *     override val keys: Set<K>
 *         get() {
 *             if (_keys == null) {
 *                 _keys = object : AbstractSet<K>() {
 *                     override operator fun contains(element: K): Boolean = containsKey(element)
 *                     override operator fun iterator(): Iterator<K> {
 *                         val entryIterator = entries.iterator()
 *                         return object : Iterator<K> {
 *                             override fun hasNext(): Boolean = entryIterator.hasNext()
 *                             override fun next(): K = entryIterator.next().key
 *                         }
 *                     }
 *                     override val size: Int get() = this@AbstractMap.size
 *                 }
 *                 @kotlin.jvm.Volatile
 *                 private var _keys: Set<K>? = null
 *             }
 *             return _keys!!
 *         }
 *     /**
 *      * Returns a read-only [Collection] of all values in this map.
 *      * Accessing this property first time creates a values view from [entries].
 *      * All subsequent accesses just return the created instance.
 *      */
 *     override val values: Collection<V>
 *         get() {
 *             if (_values == null) {
 *                 _values = object : AbstractCollection<V>() {
 *                     override operator fun contains(element: @UnsafeVariance V): Boolean = containsValue(element)
 *                     override operator fun iterator(): Iterator<V> {
 *                         val entryIterator = entries.iterator()
 *                         return object : Iterator<V> {
 *                             override fun hasNext(): Boolean = entryIterator.hasNext()
 *                             override fun next(): V = entryIterator.next().value
 *                         }
 *                     }
 *                     override val size: Int get() = this@AbstractMap.size
 *                 }
 *                 @kotlin.jvm.Volatile
 *                 private var _values: Collection<V>? = null
 *             }
 *             return _values!!
 *         }
 *     private fun implFindEntry(key: K): Map.Entry<K, V>? = entries.firstOrNull { it.key == key }
 *     internal companion object {
 *         internal fun entryHashCode(e: Map.Entry<*, *>): Int = with(e) { (key?.hashCode() ?: 0) xor (value?.hashCode() ?: 0) }
 *         internal fun entryToString(e: Map.Entry<*, *>): String = with(e) { "$key=$value" }
 *         internal fun entryEquals(e: Map.Entry<*, *>, other: Any?): Boolean {
 *             if (other !is Map.Entry<*, *>) return false
 *             return e.key == other.key && e.value == other.value
 *         }
 *     }
}
"/>
 * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.
 * Use of

```

```

this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*\npackage kotlin.collections\n\n/**\n * Provides a skeletal implementation of the read-only [Set] interface.\n *\n * This class is intended to help implementing read-only sets so it doesn't support concurrent modification tracking.\n *\n * @param E the type of elements contained in the set. The set is covariant in its element type.\n *\n @SinceKotlin("1.1")\npublic abstract class AbstractSet<out E> protected constructor() :
AbstractCollection<E>(), Set<E> {\n\n /**\n * Compares this set with other set instance with the unordered
structural equality.\n *\n * @return true, if [other] instance is a [Set] of the same size, all elements of which are
contained in this set.\n *\n override fun equals(other: Any?): Boolean {\n if (other === this) return true\n
if (other !is Set<*>) return false\n return setEquals(this, other)\n }\n\n /**\n * Returns the hash code
value for this set.\n *\n override fun hashCode(): Int = unorderedHashCode(this)\n\n internal companion
object {\n internal fun unorderedHashCode(c: Collection<*>): Int {\n var hashCode = 0\n for
(element in c) {\n hashCode += (element?.hashCode() ?: 0)\n }\n return hashCode\n
}\n\n internal fun setEquals(c: Set<*>, other: Set<*>): Boolean {\n if (c.size != other.size) return false\n
return c.containsAll(other)\n }\n }\n\n", "*/\n * Copyright 2010-2019 JetBrains s.r.o. and Kotlin
Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.\n *\npackage kotlin.collections\n\n/**\n * Resizable-array implementation
of the deque data structure.\n *\n * The name deque is short for "double ended queue" and is usually pronounced
"deck".\n *\n * The collection provide methods for convenient access to the both ends.\n * It also implements
[MutableList] interface and supports efficient get/set operations by index.\n
*\n @SinceKotlin("1.4")\n @WasExperimental(ExperimentalStdlibApi::class)\npublic class ArrayDeque<E> :
AbstractMutableList<E> {\n private var head: Int = 0\n private var elementData: Array<Any?>\n override
var size: Int = 0\n private set\n\n /**\n * Constructs an empty deque with specified [initialCapacity], or
throws [IllegalArgumentException] if [initialCapacity] is negative.\n *\n public constructor(initialCapacity:
Int) {\n elementData = when {\n initialCapacity == 0 -> emptyElementData\n initialCapacity > 0 -
> arrayOfNulls(initialCapacity)\n else -> throw IllegalArgumentException("Illegal Capacity:
$initialCapacity")\n }\n }\n\n /**\n * Constructs an empty deque.\n *\n public constructor() {\n
elementData = emptyElementData\n }\n\n /**\n * Constructs a deque that contains the same elements as the
specified [elements] collection in the same order.\n *\n public constructor(elements: Collection<E>) {\n
elementData = elements.toArray()\n size = elementData.size\n if (elementData.isEmpty())\n
elementData = emptyElementData\n }\n\n /**\n * Ensures that the capacity of this deque is at least equal to
the specified [minCapacity].\n *\n * If the current capacity is less than the [minCapacity], a new backing
storage is allocated with greater capacity.\n * Otherwise, this method takes no action and simply returns.\n *\n
private fun ensureCapacity(minCapacity: Int) {\n if (minCapacity < 0) throw IllegalStateException("Deque is
too big.") // overflow\n if (minCapacity <= elementData.size) return\n if (elementData ===
emptyElementData) {\n elementData = arrayOfNulls(minCapacity.coerceAtLeast(defaultMinCapacity))\n
return\n }\n\n val newCapacity = newCapacity(elementData.size, minCapacity)\n
copyElements(newCapacity)\n }\n\n /**\n * Creates a new array with the specified [newCapacity] size and
copies elements in the [elementData] array to it.\n *\n private fun copyElements(newCapacity: Int) {\n val
newElements = arrayOfNulls<Any?>(newCapacity)\n elementData.copyInto(newElements, 0, head,
elementData.size)\n elementData.copyInto(newElements, elementData.size - head, 0, head)\n head = 0\n
elementData = newElements\n }\n\n @kotlin.internal.InlineOnly\n private inline fun
internalGet(internalIndex: Int): E {\n @Suppress("UNCHECKED_CAST")\n return
elementData[internalIndex] as E\n }\n\n private fun positiveMod(index: Int): Int = if (index >=
elementData.size) index - elementData.size else index\n\n private fun negativeMod(index: Int): Int = if (index < 0)
index + elementData.size else index\n\n @kotlin.internal.InlineOnly\n private inline fun internalIndex(index:
Int): Int = positiveMod(head + index)\n\n private fun incremented(index: Int): Int = if (index ==
elementData.lastIndex) 0 else index + 1\n\n private fun decremented(index: Int): Int = if (index == 0)
elementData.lastIndex else index - 1\n\n override fun isEmpty(): Boolean = size == 0\n\n /**\n * Returns the

```

```

first element, or throws [NoSuchElementException] if this deque is empty.\n    */\n    public fun first(): E = if
(isEmpty()) throw NoSuchElementException("\nArrayDeque is empty.\n") else internalGet(head)\n\n    /**\n    * Returns the first element, or `null` if this deque is empty.\n    */\n    public fun firstOrNull(): E? = if (isEmpty()) null
else internalGet(head)\n\n    /**\n    * Returns the last element, or throws [NoSuchElementException] if this deque
is empty.\n    */\n    public fun last(): E = if (isEmpty()) throw NoSuchElementException("\nArrayDeque is empty.\n")
else internalGet(internalIndex(lastIndex))\n\n    /**\n    * Returns the last element, or `null` if this deque is empty.\n    */\n    public fun lastOrNull(): E? = if (isEmpty()) null else internalGet(internalIndex(lastIndex))\n\n    /**\n    * Prepends the specified [element] to this deque.\n    */\n    public fun addFirst(element: E) {\n        ensureCapacity(size + 1)\n        head = decremented(head)\n        elementData[head] = element\n        size += 1\n    }\n\n    /**\n    * Appends the specified [element] to this deque.\n    */\n    public fun addLast(element: E) {\n        ensureCapacity(size + 1)\n        elementData[internalIndex(size)] = element\n        size += 1\n    }\n\n    /**\n    * Removes the first element from this deque and returns that removed element, or throws [NoSuchElementException]
if this deque is empty.\n    */\n    public fun removeFirst(): E {\n        if (isEmpty()) throw
NoSuchElementException("\nArrayDeque is empty.\n")\n        val element = internalGet(head)\n        elementData[head] = null\n        head = incremented(head)\n        size -= 1\n        return element\n    }\n\n    /**\n    * Removes the first element from this deque and returns that removed element, or returns `null` if this deque is
empty.\n    */\n    public fun removeFirstOrNull(): E? = if (isEmpty()) null else removeFirst()\n\n    /**\n    * Removes the last element from this deque and returns that removed element, or throws [NoSuchElementException]
if this deque is empty.\n    */\n    public fun removeLast(): E {\n        if (isEmpty()) throw
NoSuchElementException("\nArrayDeque is empty.\n")\n        val internalLastIndex = internalIndex(lastIndex)\n        val element = internalGet(internalLastIndex)\n        elementData[internalLastIndex] = null\n        size -= 1\n        return element\n    }\n\n    /**\n    * Removes the last element from this deque and returns that removed element, or
returns `null` if this deque is empty.\n    */\n    public fun removeLastOrNull(): E? = if (isEmpty()) null else
removeLast()\n\n    // MutableList, MutableCollection\n    public override fun add(element: E): Boolean {\n        addLast(element)\n        return true\n    }\n\n    public override fun add(index: Int, element: E) {\n        AbstractList.checkPositionIndex(index, size)\n        if (index == size) {\n            addLast(element)\n            return\n        } else if (index == 0) {\n            addFirst(element)\n            return\n        }\n        ensureCapacity(size
+ 1)\n        // Elements in circular array lay in 2 ways:\n        // 1. `head` is less than `tail`:    [#, #, e1, e2, e3,
#]\n        // 2. `head` is greater than `tail`:    [e3, #, #, #, e1, e2]\n        // where head is the index of the first element
in the circular array,\n        // and tail is the index following the last element.\n        // At this point the
insertion index is not equal to head or tail.\n        // Also the circular array can store at least one more element.\n        // Depending on where the given element must be inserted the preceding or the succeeding\n        // elements
will be shifted to make room for the element to be inserted.\n        // In case the preceding elements are
shifted:\n        // * if the insertion index is greater than the head (regardless of circular array form)\n        // ->
shift the preceding elements\n        // * otherwise, the circular array has (2) form and the insertion index is less than
tail\n        // -> shift all elements in the back of the array\n        // -> shift preceding elements in the front of the
array\n        // In case the succeeding elements are shifted:\n        // * if the insertion index is less than the tail
(regardless of circular array form)\n        // -> shift the succeeding elements\n        // * otherwise, the circular
array has (2) form and the insertion index is greater than head\n        // -> shift all elements in the front of the
array\n        // -> shift succeeding elements in the back of the array\n        val internalIndex =
internalIndex(index)\n        if (index < (size + 1) shr 1) {\n            // closer to the first element -> shift preceding
elements\n            val decrementedInternalIndex = decremented(internalIndex)\n            val decrementedHead =
decremented(head)\n            if (decrementedInternalIndex >= head) {\n                elementData[decrementedHead]
= elementData[head] // head can be zero\n                elementData.copyInto(elementData, head, head + 1,
decrementedInternalIndex + 1)\n            } else { // head > tail\n                elementData.copyInto(elementData, head -
1, head, elementData.size) // head can't be zero\n                elementData[elementData.size - 1] = elementData[0]\n                elementData.copyInto(elementData, 0, 1, decrementedInternalIndex + 1)\n            }\n            elementData[decrementedInternalIndex] = element\n            head = decrementedHead\n        } else {\n            //

```

```

closer to the last element -> shift succeeding elements\n        val tail = internalIndex(size)\n\n        if
(internalIndex < tail) {\n            elementData.copyInto(elementData, internalIndex + 1, internalIndex, tail)\n
} else { // head > tail\n        elementData.copyInto(elementData, 1, 0, tail)\n        elementData[0] =
elementData[elementData.size - 1]\n        elementData.copyInto(elementData, internalIndex + 1, internalIndex,
elementData.size - 1)\n    }\n    elementData[internalIndex] = element\n    }\n    size += 1\n    }\n\nprivate fun copyCollectionElements(internalIndex: Int, elements: Collection<E>) {\n    val iterator =
elements.iterator()\n    for (index in internalIndex until elementData.size) {\n        if (!iterator.hasNext())
break\n        elementData[index] = iterator.next()\n    }\n    for (index in 0 until head) {\n        if
(!iterator.hasNext()) break\n        elementData[index] = iterator.next()\n    }\n    size += elements.size\n
}\n\n    public override fun addAll(elements: Collection<E>): Boolean {\n        if (elements.isEmpty()) return false\n
        ensureCapacity(this.size + elements.size)\n        copyCollectionElements(internalIndex(size), elements)\n
return true\n    }\n\n    public override fun addAll(index: Int, elements: Collection<E>): Boolean {\n
AbstractList.checkPositionIndex(index, size)\n\n        if (elements.isEmpty()) {\n            return false\n        } else if
(index == size) {\n            return addAll(elements)\n        }\n        ensureCapacity(this.size + elements.size)\n
val tail = internalIndex(size)\n        val internalIndex = internalIndex(index)\n        val elementsSize =
elements.size\n\n        if (index < (size + 1) shr 1) {\n            // closer to the first element -> shift preceding
elements\n\n            var shiftedHead = head - elementsSize\n\n            if (internalIndex >= head) {\n                if
(shiftedHead >= 0) {\n                    elementData.copyInto(elementData, shiftedHead, head, internalIndex)\n
} else { // head < tail, insertion leads to head >= tail\n                shiftedHead += elementData.size\n                val
elementsToShift = internalIndex - head\n                val shiftToBack = elementData.size - shiftedHead\n\n                if (shiftToBack >= elementsToShift) {\n                    elementData.copyInto(elementData, shiftedHead, head,
internalIndex)\n                } else {\n                    elementData.copyInto(elementData, shiftedHead, head, head +
shiftToBack)\n                }\n                elementData.copyInto(elementData, 0, head + shiftToBack, internalIndex)\n
            }\n        }\n    } else { // head > tail, internalIndex < tail\n        elementData.copyInto(elementData,
shiftedHead, head, elementData.size)\n        if (elementsSize >= internalIndex) {\n            elementData.copyInto(elementData, elementData.size - elementsSize, 0, internalIndex)\n        } else {\n            elementData.copyInto(elementData, elementData.size - elementsSize, 0, elementsSize)\n
            elementData.copyInto(elementData, 0, elementsSize, internalIndex)\n        }\n    }\n    head =
shiftedHead\n    copyCollectionElements(negativeMod(internalIndex - elementsSize), elements)\n    } else
{\n        // closer to the last element -> shift succeeding elements\n        val shiftedInternalIndex =
internalIndex + elementsSize\n\n        if (internalIndex < tail) {\n            if (tail + elementsSize <=
elementData.size) {\n                elementData.copyInto(elementData, shiftedInternalIndex, internalIndex, tail)\n
            } else { // head < tail, insertion leads to head >= tail\n                if (shiftedInternalIndex >= elementData.size)
{\n                    elementData.copyInto(elementData, shiftedInternalIndex - elementData.size, internalIndex, tail)\n
                }\n            } else {\n                val shiftToFront = tail + elementsSize - elementData.size\n
                elementData.copyInto(elementData, 0, tail - shiftToFront, tail)\n                elementData.copyInto(elementData,
shiftedInternalIndex, internalIndex, tail - shiftToFront)\n            }\n        }\n    } else { // head > tail,
internalIndex > head\n        elementData.copyInto(elementData, elementsSize, 0, tail)\n        if
(shiftedInternalIndex >= elementData.size) {\n            elementData.copyInto(elementData, shiftedInternalIndex
- elementData.size, internalIndex, elementData.size)\n        } else {\n            elementData.copyInto(elementData, 0, elementData.size - elementsSize, elementData.size)\n
            elementData.copyInto(elementData, shiftedInternalIndex, internalIndex, elementData.size - elementsSize)\n
        }\n    }\n    }\n    copyCollectionElements(internalIndex, elements)\n    }\n\n    return true\n    }\n\n    public
override fun get(index: Int): E {\n        AbstractList.checkElementIndex(index, size)\n        return
internalGet(internalIndex(index))\n    }\n\n    public override fun set(index: Int, element: E): E {\n
AbstractList.checkElementIndex(index, size)\n        val internalIndex = internalIndex(index)\n        val oldElement
= internalGet(internalIndex)\n        elementData[internalIndex] = element\n        return oldElement\n    }\n\n    public
override fun contains(element: E): Boolean = indexOf(element) != -1\n\n    public override fun

```

```

indexOf(element: E): Int {\n    val tail = internalIndex(size)\n\n    if (head < tail) {\n        for (index in head
until tail) {\n            if (element == elementData[index]) return index - head\n        }\n    } else if (head >=
tail) {\n        for (index in head until elementData.size) {\n            if (element == elementData[index]) return
index - head\n        }\n    } for (index in 0 until tail) {\n        if (element == elementData[index]) return
index + elementData.size - head\n    }\n\n    return -1\n }\n\n public override fun
lastIndexOf(element: E): Int {\n    val tail = internalIndex(size)\n\n    if (head < tail) {\n        for (index in tail
- 1 downTo head) {\n            if (element == elementData[index]) return index - head\n        }\n    } else if
(head > tail) {\n        for (index in tail - 1 downTo 0) {\n            if (element == elementData[index]) return
index + elementData.size - head\n        }\n    } for (index in elementData.lastIndex downTo head) {\n
if (element == elementData[index]) return index - head\n    }\n\n    return -1\n }\n\n public
override fun remove(element: E): Boolean {\n    val index = indexOf(element)\n    if (index == -1) return
false\n    removeAt(index)\n    return true\n }\n\n public override fun removeAt(index: Int): E {\n
AbstractList.checkElementIndex(index, size)\n\n    if (index == lastIndex) {\n        return removeLast()\n    }
else if (index == 0) {\n        return removeFirst()\n    }\n\n    val internalIndex = internalIndex(index)\n
val element = internalGet(internalIndex)\n\n    if (index < size shr 1) {\n        // closer to the first element ->
shift preceding elements\n        if (internalIndex >= head) {\n            elementData.copyInto(elementData, head
+ 1, head, internalIndex)\n        } else { // head > tail, internalIndex < head\n
elementData.copyInto(elementData, 1, 0, internalIndex)\n            elementData[0] = elementData[elementData.size
- 1]\n            elementData.copyInto(elementData, head + 1, head, elementData.size - 1)\n        }\n    }\n
elementData[head] = null\n    head = incremented(head)\n    } else {\n        // closer to the last element ->
shift succeeding elements\n        val internalLastIndex = internalIndex(lastIndex)\n\n        if (internalIndex <=
internalLastIndex) {\n            elementData.copyInto(elementData, internalIndex, internalIndex + 1,
internalLastIndex + 1)\n        } else { // head > tail, internalIndex > head\n
elementData.copyInto(elementData, internalIndex, internalIndex + 1, elementData.size)\n
elementData[elementData.size - 1] = elementData[0]\n            elementData.copyInto(elementData, 0, 1,
internalLastIndex + 1)\n        }\n    }\n    elementData[internalLastIndex] = null\n    }\n    size -= 1\n\n
return element\n }\n\n public override fun removeAll(elements: Collection<E>): Boolean = filterInPlace {
!elements.contains(it) }\n\n public override fun retainAll(elements: Collection<E>): Boolean = filterInPlace {
elements.contains(it) }\n\n private inline fun filterInPlace(predicate: (E) -> Boolean): Boolean {\n    if
(this.isEmpty() || elementData.isEmpty())\n        return false\n\n    val tail = internalIndex(size)\n    var
newTail = head\n    var modified = false\n\n    if (head < tail) {\n        for (index in head until tail) {\n
val element = elementData[index]\n\n            @Suppress("UNCHECKED_CAST")\n            if
(predicate(element as E))\n                elementData[newTail++] = element\n            else\n                modified =
true\n        }\n    }\n    elementData.fill(null, newTail, tail)\n    } else {\n        for (index in head until
elementData.size) {\n            val element = elementData[index]\n            elementData[index] = null\n\n
@Suppress("UNCHECKED_CAST")\n            if (predicate(element as E))\n                elementData[newTail++] = element\n            else\n                modified = true\n        }\n    }\n    newTail =
positiveMod(newTail)\n\n    for (index in 0 until tail) {\n        val element = elementData[index]\n
elementData[index] = null\n\n        @Suppress("UNCHECKED_CAST")\n        if (predicate(element as
E)) {\n            elementData[newTail] = element\n            newTail = incremented(newTail)\n        }
else {\n            modified = true\n        }\n    }\n    }\n    if (modified)\n        size =
negativeMod(newTail - head)\n\n    return modified\n }\n\n public override fun clear() {\n    val tail =
internalIndex(size)\n    if (head < tail) {\n        elementData.fill(null, head, tail)\n    } else if (isEmpty())
{\n        elementData.fill(null, head, elementData.size)\n        elementData.fill(null, 0, tail)\n    }\n    head =
0\n    size = 0\n    }\n\n    @Suppress("NOTHING_TO_OVERRIDE")\n    override fun <T> toArray(array:
Array<T>): Array<T> {\n        @Suppress("UNCHECKED_CAST")\n        val dest = (if (array.size >= size) array
else arrayOfNulls(array, size)) as Array<Any?>\n\n        val tail = internalIndex(size)\n        if (head < tail) {\n
elementData.copyInto(dest, startIndex = head, endIndex = tail)\n        } else if (isEmpty()) {\n

```

```

elementData.copyInto(dest, destinationOffset = 0, startIndex = head, endIndex = elementData.size)\n
elementData.copyInto(dest, destinationOffset = elementData.size - head, startIndex = 0, endIndex = tail)\n    }\n
    if (dest.size > size) {\n        dest[size] = null // null-terminate\n    }\n\n
@Suppress("UNCHECKED_CAST")\n    return dest as Array<T>\n }\n\n
@Suppress("NOTHING_TO_OVERRIDE")\n    override fun toArray(): Array<Any?> {\n        return
toArray(arrayOfNulls<Any?>(size))\n    }\n\n // for testing\n    internal fun <T> testToArray(array: Array<T>):
Array<T> = toArray(array)\n    internal fun testToArray(): Array<Any?> = toArray()\n\n    internal companion
object {\n        private val emptyElementData = emptyArray<Any?>()\n        private const val maxArraySize =
Int.MAX_VALUE - 8\n        private const val defaultMinCapacity = 10\n        internal fun
newCapacity(oldCapacity: Int, minCapacity: Int): Int {\n            // overflow-conscious\n            var newCapacity =
oldCapacity + (oldCapacity shr 1)\n            if (newCapacity - minCapacity < 0)\n                newCapacity =
minCapacity\n            if (newCapacity - maxArraySize > 0)\n                newCapacity = if (minCapacity >
maxArraySize) Int.MAX_VALUE else maxArraySize\n            return newCapacity\n        }\n    }\n\n // For testing
only\n    internal fun internalStructure(structure: (head: Int, elements: Array<Any?>) -> Unit) {\n        val tail =
internalIndex(size)\n        val head = if (isEmpty() || head < tail) head else head - elementData.size\n
structure(head, toArray())\n    }\n}", "/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n
*/\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("ArraysKt")\n\npackage
kotlin.collections\n\nimport kotlin.contracts.*\n\n/**\n * Returns a single list of all elements from all arrays in the
given array.\n * @sample samples.collections.Arrays.Transformations.flattenArray\n */\npublic fun <T> Array<out
Array<out T>>.flatten(): List<T> {\n    val result = ArrayList<T>(sumOf { it.size })\n    for (element in this) {\n
result.addAll(element)\n    }\n    return result\n}\n\n/**\n * Returns a pair of lists, where\n * *first* list is built from
the first values of each pair from this array,\n * *second* list is built from the second values of each pair from this
array.\n * @sample samples.collections.Arrays.Transformations.unzipArray\n */\npublic fun <T, R> Array<out
Pair<T, R>>.unzip(): Pair<List<T>, List<R>> {\n    val listT = ArrayList<T>(size)\n    val listR =
ArrayList<R>(size)\n    for (pair in this) {\n        listT.add(pair.first)\n        listR.add(pair.second)\n    }\n    return
listT to listR\n}\n\n/**\n * Returns `true` if this nullable array is either null or empty.\n * @sample
samples.collections.Arrays.Usage.arrayIsNullOrEmpty\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline fun Array<*>?.isNullOrEmpty(): Boolean
{\n    contract {\n        returns(false) implies (this@isNullOrEmpty != null)\n    }\n    return this == null ||
this.isEmpty()\n}\n\n/**\n * Returns this array if it's not empty\n * or the result of calling [defaultValue] function if
the array is empty.\n * @sample samples.collections.Arrays.Usage.arrayIfEmpty\n
*/\n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\n@Suppress("UPPER_BOUND_CANNOT_BE_ARRAY")\npublic inline fun <C, R> C.ifEmpty(defaultValue: () -> R): R where C : Array<*>, C : R =\n    if (isEmpty())
defaultValue() else
this\n\n\n@OptIn(ExperimentalUnsignedTypes::class)\n@SinceKotlin("1.3")\n@PublishedApi\n@kotlin.jvm.Jvm
Name("contentDeepEquals")\n@kotlin.js.JsName("contentDeepEqualsImpl")\ninternal fun <T> Array<out
T?>.contentDeepEqualsImpl(other: Array<out T?>): Boolean {\n    if (this === other) return true\n    if (this == null
|| other == null || this.size != other.size) return false\n\n    for (i in indices) {\n        val v1 = this[i]\n        val v2 =
other[i]\n\n        if (v1 === v2) {\n            continue\n        } else if (v1 == null || v2 == null) {\n            return false\n
        }\n        when {\n            v1 is Array<*> && v2 is Array<*> -> if (!v1.contentDeepEquals(v2)) return
false\n            v1 is ByteArray && v2 is ByteArray -> if (!v1.contentEquals(v2)) return false\n            v1 is
ShortArray && v2 is ShortArray -> if (!v1.contentEquals(v2)) return false\n            v1 is IntArray && v2 is
IntArray -> if (!v1.contentEquals(v2)) return false\n            v1 is LongArray && v2 is LongArray -> if
(!v1.contentEquals(v2)) return false\n            v1 is FloatArray && v2 is FloatArray -> if (!v1.contentEquals(v2))
return false\n            v1 is DoubleArray && v2 is DoubleArray -> if (!v1.contentEquals(v2)) return false\n
            v1 is CharArray && v2 is CharArray -> if (!v1.contentEquals(v2)) return false\n            v1 is BooleanArray &&

```

```

v2 is BooleanArray -> if (!v1.contentEquals(v2)) return false\n\n      v1 is UByteArray && v2 is UByteArray
-> if (!v1.contentEquals(v2)) return false\n      v1 is UShortArray && v2 is UShortArray -> if
(!v1.contentEquals(v2)) return false\n      v1 is UIntArray && v2 is UIntArray -> if (!v1.contentEquals(v2))
return false\n      v1 is ULongArray && v2 is ULongArray -> if (!v1.contentEquals(v2)) return false\n\n
else -> if (v1 != v2) return false\n    }\n\n    return
true\n}\n\n@SinceKotlin("1.3")\n@PublishedApi\n@kotlin.jvm.JvmName("contentDeepToString")\n@kotlin.js.
JsName("contentDeepToStringImpl")\ninternal fun <T> Array<out T>?.contentDeepToStringImpl(): String {\n
if (this == null) return "null"\n    val length = size.coerceAtMost((Int.MAX_VALUE - 2) / 5) * 5 + 2 // in order not
to overflow Int.MAX_VALUE\n    return buildString(length) {\n        contentDeepToStringInternal(this,
mutableListOf())\n    }\n}\n\n@OptIn(ExperimentalUnsignedTypes::class)\nprivate fun <T> Array<out
T>.contentDeepToStringInternal(result: StringBuilder, processed: MutableList<Array<*>>) {\n    if (this in
processed) {\n        result.append("[...]")\n        return\n    }\n    processed.add(this)\n    result.append('[')\n\n    for (i
in indices) {\n        if (i != 0) {\n            result.append(", ")\n        }\n        val element = this[i]\n        when
(element) {\n            null -> result.append("null")\n            is Array<*> ->
element.contentDeepToStringInternal(result, processed)\n            is ByteArray ->
result.append(element.contentToString())\n            is ShortArray -> result.append(element.contentToString())\n
            is IntArray -> result.append(element.contentToString())\n            is LongArray ->
result.append(element.contentToString())\n            is FloatArray -> result.append(element.contentToString())\n
            is DoubleArray -> result.append(element.contentToString())\n            is CharArray ->
result.append(element.contentToString())\n            is BooleanArray -> result.append(element.contentToString())\n
            is UByteArray -> result.append(element.contentToString())\n            is UShortArray ->
result.append(element.contentToString())\n            is UIntArray -> result.append(element.contentToString())\n
            is ULongArray -> result.append(element.contentToString())\n            else ->
result.append(element.toString())\n        }\n    }\n    result.append(',')\n    processed.removeAt(processed.lastIndex)\n}"/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\npackage kotlin.collections\n\n/** Returns true if the brittle contains optimization
is enabled. See KT-45438. */\ninternal expect fun brittleContainsOptimizationEnabled(): Boolean\n\n/**\n * Returns true if [brittleContainsOptimizationEnabled] is true\n * and it's safe to convert this collection to a set
without changing contains method behavior.\n */\nprivate fun <T> Collection<T>.safeToConvertToSet() =
brittleContainsOptimizationEnabled() && size > 2 && this is ArrayList\n\n/**\n * When [brittleContainsOptimizationEnabled] is true:\n * - Converts this [Iterable] to a set if it is not a [Collection].\n * -
Converts this [Collection] to a set, when it's worth so and it doesn't change contains method behavior.\n * -
Otherwise returns this.\n * When [brittleContainsOptimizationEnabled] is false:\n * - Converts this [Iterable] to a
list if it is not a [Collection].\n * - Otherwise returns this.\n */\ninternal fun <T>
Iterable<T>.convertToSetForSetOperationWith(source: Iterable<T>): Collection<T> =\n    when (this) {\n        is
Set -> this\n        is Collection -> when {\n            source is Collection && source.size < 2 -> this\n
            else -> if (this.safeToConvertToSet()) toHashSet() else this\n        }\n        else -> if
(brittleContainsOptimizationEnabled()) toHashSet() else toList()\n    }\n\n/**\n * When [brittleContainsOptimizationEnabled] is true:\n * - Converts this [Iterable] to a set if it is not a [Collection].\n * -
Converts this [Collection] to a set, when it's worth so and it doesn't change contains method behavior.\n * -
Otherwise returns this.\n * When [brittleContainsOptimizationEnabled] is false:\n * - Converts this [Iterable] to a
list if it is not a [Collection].\n * - Otherwise returns this.\n */\ninternal fun <T>
Iterable<T>.convertToSetForSetOperation(): Collection<T> =\n    when (this) {\n        is Set -> this\n        is
Collection -> if (this.safeToConvertToSet()) toHashSet() else this\n        else -> if
(brittleContainsOptimizationEnabled()) toHashSet() else toList()\n    }\n\n/**\n * Converts this sequence to a set if
[brittleContainsOptimizationEnabled] is true,\n * otherwise converts it to a list.\n */\ninternal fun <T>
Sequence<T>.convertToSetForSetOperation(): Collection<T> =\n    if (brittleContainsOptimizationEnabled())

```

```

toHashSet() else toList()\n\n/**\n * Converts this array to a set if [brittleContainsOptimizationEnabled] is true,\n * otherwise converts it to a list.\n */\ninternal fun <T> Array<T>.convertToSetForSetOperation(): Collection<T> =\n    if (brittleContainsOptimizationEnabled()) toHashSet() else asList()", /*\n * Copyright 2010-2018 JetBrains s.r.o.\n * and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license\n * that can be found in the license/LICENSE.txt file.\n */\npackage kotlin.collections\n\n/**\n * Data class\n * representing a value from a collection or sequence, along with its index in that collection or sequence.\n */\n * @property value the underlying value.\n * @property index the index of the value in the collection or sequence.\n */\npublic data class IndexedValue<out T>(public val index: Int, public val value: T)\n", /*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the\n * Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n@file:kotlin.jvm.JvmName("MapAccessorsKt")\npackage kotlin.collections\n\nimport\nkotlin.reflect.KProperty\nimport kotlin.internal.Exact\n\n/**\n * Returns the value of the property for the given\n * object from this read-only map.\n * @param thisRef the object for which the value is requested (not used).\n * @param property the metadata for the property, used to get the name of property and lookup the value\n * corresponding to this name in the map.\n * @return the property value.\n */\n * @throws NoSuchElementException\n * when the map doesn't contain value for the property name and doesn't provide an implicit default (see\n * [withDefault]).\n */\n@kotlin.internal.InlineOnly\npublic inline operator fun <V, V1 : V> Map<in String, @Exact\nV>.getValue(thisRef: Any?, property: KProperty<*>): V1 =\n    @Suppress("UNCHECKED_CAST")\n    (getOrNull(thisRef, property.name) as V1)\n\n/**\n * Returns the value of the property for the given object from\n * this mutable map.\n * @param thisRef the object for which the value is requested (not used).\n * @param property the metadata for the property, used to get the name of property and lookup the value corresponding to this name in\n * the map.\n * @return the property value.\n */\n * @throws NoSuchElementException\n * when the map doesn't contain\n * value for the property name and doesn't provide an implicit default (see [withDefault]).\n */\n@kotlin.jvm.JvmName("getVar")\n@kotlin.internal.InlineOnly\npublic inline operator fun <V, V1 : V>\nMutableMap<in String, out @Exact V>.getValue(thisRef: Any?, property: KProperty<*>): V1 =\n    @Suppress("UNCHECKED_CAST")\n    (getOrNull(thisRef, property.name) as V1)\n\n/**\n * Stores the value of\n * the property for the given object in this mutable map.\n * @param thisRef the object for which the value is\n * requested (not used).\n * @param property the metadata for the property, used to get the name of property and store\n * the value associated with that name in the map.\n * @param value the value to set.\n */\n@kotlin.internal.InlineOnly\npublic inline operator fun <V> MutableMap<in String, in V>.setValue(thisRef:\nAny?, property: KProperty<*>, value: V) {\n    this.put(property.name, value)\n}\n", /*\n * Copyright 2010-2018\n * JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the\n * Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("MapsKt")\npackage\nkotlin.collections\n\n/**\n * Returns the value for the given key, or the implicit default value for this map.\n * By\n * default no implicit value is provided for maps and a [NoSuchElementException] is thrown.\n * To create a map with\n * implicit default value use [withDefault] method.\n */\n * @throws NoSuchElementException\n * when the map doesn't\n * contain a value for the specified key and no implicit default was provided for that map.\n */\n@kotlin.jvm.JvmName("getOrNull")\n@PublishedApi\ninternal fun <K, V> Map<K,\nV>.getOrNull(key: K): V {\n    if (this is MapWithDefault)\n        return\n        this.getOrNull(key)\n    return getOrNull(key, { throw NoSuchElementException("Key $key\nis missing in the map.") })\n}\n\n/**\n * Returns a wrapper of this read-only map, having the implicit default value\n * provided with the specified function [defaultValue].\n */\n * This implicit default value is used when the original\n * map doesn't contain a value for the key specified\n * and a value is obtained with [Map.getValue] function, for\n * example when properties are delegated to the map.\n */\n * When this map already has an implicit default value\n * provided with a former call to [withDefault], it is being replaced by this call.\n */\npublic fun <K, V> Map<K,\nV>.withDefault(defaultValue: (key: K) -> V): Map<K, V> =\n    when (this) {\n        is MapWithDefault ->\n            this.withDefault(defaultValue)\n        else -> MapWithDefaultImpl(this, defaultValue)\n    }\n\n/**\n * Returns

```

a wrapper of this mutable map, having the implicit default value provided with the specified function [defaultValue].

\* This implicit default value is used when the original map doesn't contain a value for the key specified and a value is obtained with [Map.getValue] function, for example when properties are delegated to the map.

\* When this map already has an implicit default value provided with a former call to [withDefault], it is being replaced by this call.

```

@kotlin.jvm.JvmName("withDefaultMutable")
public fun <K, V>
MutableMap<K, V>.withDefault(defaultValue: (key: K) -> V): MutableMap<K, V> =
    when (this) {
        is MutableMapWithDefault -> this.map.withDefault(defaultValue)
        else -> MutableMapWithDefaultImpl(this,
            defaultValue)
    }

private interface MapWithDefault<K, out V> : Map<K, V> {
    public val map: Map<K, V>
    public fun getOrElse(key: K): V
}

private interface MutableMapWithDefault<K, V> :
MutableMap<K, V>, MapWithDefault<K, V> {
    public override val map: MutableMap<K, V>
}

private class MapWithDefaultImpl<K, out V>(public override val map: Map<K, V>, private val default: (key: K) -> V) :
MapWithDefault<K, V> {
    override fun equals(other: Any?): Boolean = map.equals(other)
    override fun hashCode(): Int = map.hashCode()
    override fun toString(): String = map.toString()
    override val size: Int get() = map.size
    override fun isEmpty(): Boolean = map.isEmpty()
    override fun containsKey(key: K): Boolean = map.containsKey(key)
    override fun containsValue(value: @UnsafeVariance V): Boolean = map.containsValue(value)
    override fun get(key: K): V? = map.get(key)
    override val keys: Set<K> get() = map.keys
    override val values: Collection<V> get() = map.values
    override val entries: Set<Map.Entry<K, V>> get() = map.entries
    override fun getOrElse(key: K): V = map.getOrElse(key, { default(key) })
}

private class MutableMapWithDefaultImpl<K, V>(public override val map: MutableMap<K, V>, private val default: (key: K) -> V) : MutableMapWithDefault<K, V> {
    override fun equals(other: Any?): Boolean = map.equals(other)
    override fun hashCode(): Int = map.hashCode()
    override fun toString(): String = map.toString()
    override val size: Int get() = map.size
    override fun isEmpty(): Boolean = map.isEmpty()
    override fun containsKey(key: K): Boolean = map.containsKey(key)
    override fun containsValue(value: @UnsafeVariance V): Boolean = map.containsValue(value)
    override fun get(key: K): V? = map.get(key)
    override val keys: MutableSet<K> get() = map.keys
    override val values: MutableCollection<V> get() = map.values
    override val entries: MutableSet<MutableMap.MutableEntry<K, V>> get() = map.entries
    override fun put(key: K, value: V): V? = map.put(key, value)
    override fun remove(key: K): V? = map.remove(key)
    override fun putAll(from: Map<out K, V>) = map.putAll(from)
    override fun clear() = map.clear()
    override fun getOrElse(key: K): V = map.getOrElse(key, { default(key) })
}

```

\*/ \* Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors. Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.

```

@file:kotlin.jvm.JvmMultifileClass
@file:kotlin.jvm.JvmName("CollectionsKt")
package
kotlin.collections
import kotlin.random.Random
/**
 * Removes a single instance of the specified element
 * from this collection, if it is present.
 * Allows to overcome type-safety restriction of `remove` that requires
 * to pass an element of type `E`.
 * @return `true` if the element has been successfully removed; `false` if it was
 * not present in the collection.
 */
@kotlin.internal.InlineOnly
public inline fun <@kotlin.internal.OnlyInputTypes
T> MutableCollection<out T>.remove(element: T): Boolean =
    @Suppress("UNCHECKED_CAST") (this as MutableCollection<T>).remove(element)

/**
 * Removes all of this collection's elements that are also
 * contained in the specified collection.
 * Allows to overcome type-safety restriction of `removeAll` that requires
 * to pass a collection of type `Collection<E>`.
 * @return `true` if any of the specified elements was removed
 * from the collection, `false` if the collection was not modified.
 */
@kotlin.internal.InlineOnly
public inline fun <@kotlin.internal.OnlyInputTypes T> MutableCollection<out T>.removeAll(elements: Collection<T>): Boolean =
    @Suppress("UNCHECKED_CAST") (this as MutableCollection<T>).removeAll(elements)

/**
 * Retains only the elements in this collection that are contained in the specified collection.
 * Allows to overcome type-safety restriction of `retainAll` that requires to pass a collection of type `Collection<E>`.
 * @return `true` if any element was removed from the collection, `false` if the collection was not modified.
 */
@kotlin.internal.InlineOnly
public inline fun <@kotlin.internal.OnlyInputTypes T> MutableCollection<out T>.retainAll(elements: Collection<T>): Boolean =
    @Suppress("UNCHECKED_CAST") (this as

```

```

MutableCollection<T>.retainAll(elements)\n\n/**\n * Adds the specified [element] to this mutable collection.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun <T> MutableCollection<in T>.plusAssign(element: T)
{\n    this.add(element)\n}\n\n/**\n * Adds all elements of the given [elements] collection to this mutable
collection.\n *\n@kotlin.internal.InlineOnly\npublic inline operator fun <T> MutableCollection<in
T>.plusAssign(elements: Iterable<T>) {\n    this.addAll(elements)\n}\n\n/**\n * Adds all elements of the given
[elements] array to this mutable collection.\n *\n@kotlin.internal.InlineOnly\npublic inline operator fun <T>
MutableCollection<in T>.plusAssign(elements: Array<T>) {\n    this.addAll(elements)\n}\n\n/**\n * Adds all
elements of the given [elements] sequence to this mutable collection.\n *\n@kotlin.internal.InlineOnly\npublic
inline operator fun <T> MutableCollection<in T>.plusAssign(elements: Sequence<T>) {\n
this.addAll(elements)\n}\n\n/**\n * Removes a single instance of the specified [element] from this mutable
collection.\n *\n@kotlin.internal.InlineOnly\npublic inline operator fun <T> MutableCollection<in
T>.minusAssign(element: T) {\n    this.remove(element)\n}\n\n/**\n * Removes all elements contained in the given
[elements] collection from this mutable collection.\n *\n@kotlin.internal.InlineOnly\npublic inline operator fun
<T> MutableCollection<in T>.minusAssign(elements: Iterable<T>) {\n    this.removeAll(elements)\n}\n\n/**\n *
Removes all elements contained in the given [elements] array from this mutable collection.\n
*\n@kotlin.internal.InlineOnly\npublic inline operator fun <T> MutableCollection<in T>.minusAssign(elements:
Array<T>) {\n    this.removeAll(elements)\n}\n\n/**\n * Removes all elements contained in the given [elements]
sequence from this mutable collection.\n *\n@kotlin.internal.InlineOnly\npublic inline operator fun <T>
MutableCollection<in T>.minusAssign(elements: Sequence<T>) {\n    this.removeAll(elements)\n}\n\n/**\n * Adds
all elements of the given [elements] collection to this [MutableCollection].\n *\npublic fun <T>
MutableCollection<in T>.addAll(elements: Iterable<T>): Boolean {\n    when (elements) {\n        is Collection ->
return addAll(elements)\n        else -> {\n            var result: Boolean = false\n            for (item in elements)\n
if (add(item)) result = true\n            return result\n        }\n    }\n}\n\n/**\n * Adds all elements of the given
[elements] sequence to this [MutableCollection].\n *\npublic fun <T> MutableCollection<in T>.addAll(elements:
Sequence<T>): Boolean {\n    var result: Boolean = false\n    for (item in elements) {\n        if (add(item)) result =
true\n    }\n    return result\n}\n\n/**\n * Adds all elements of the given [elements] array to this
[MutableCollection].\n *\npublic fun <T> MutableCollection<in T>.addAll(elements: Array<out T>): Boolean {\n
return addAll(elements.asList())\n}\n\n/**\n * Removes all elements from this [MutableCollection] that are also
contained in the given [elements] collection.\n *\npublic fun <T> MutableCollection<in T>.removeAll(elements:
Iterable<T>): Boolean {\n    return removeAll(elements.convertToSetForSetOperationWith(this))\n}\n\n/**\n *
Removes all elements from this [MutableCollection] that are also contained in the given [elements] sequence.\n
*\npublic fun <T> MutableCollection<in T>.removeAll(elements: Sequence<T>): Boolean {\n    val set =
elements.convertToSetForSetOperation()\n    return set.isNotEmpty() && removeAll(set)\n}\n\n/**\n * Removes all
elements from this [MutableCollection] that are also contained in the given [elements] array.\n *\npublic fun <T>
MutableCollection<in T>.removeAll(elements: Array<out T>): Boolean {\n    return elements.isNotEmpty() &&
removeAll(elements.convertToSetForSetOperation())\n}\n\n/**\n * Retains only elements of this
[MutableCollection] that are contained in the given [elements] collection.\n *\npublic fun <T>
MutableCollection<in T>.retainAll(elements: Iterable<T>): Boolean {\n    return
retainAll(elements.convertToSetForSetOperationWith(this))\n}\n\n/**\n * Retains only elements of this
[MutableCollection] that are contained in the given [elements] array.\n *\npublic fun <T> MutableCollection<in
T>.retainAll(elements: Array<out T>): Boolean {\n    if (elements.isNotEmpty())\n        return
retainAll(elements.convertToSetForSetOperation())\n    else\n        return retainNothing()\n}\n\n/**\n * Retains only
elements of this [MutableCollection] that are contained in the given [elements] sequence.\n *\npublic fun <T>
MutableCollection<in T>.retainAll(elements: Sequence<T>): Boolean {\n    val set =
elements.convertToSetForSetOperation()\n    if (set.isNotEmpty())\n        return retainAll(set)\n    else\n        return
retainNothing()\n}\n\nprivate fun MutableCollection<*>.retainNothing(): Boolean {\n    val result = isEmpty()\n
clear()\n    return result\n}\n\n/**\n * Removes all elements from this [MutableIterable] that match the given
[predicate].\n *\n * @return `true` if any element was removed from this collection, or `false` when no elements

```

```

were removed and collection was not modified.\n */\npublic fun <T> MutableIterable<T>.removeAll(predicate: (T)
-> Boolean): Boolean = filterInPlace(predicate, true)\n\n/**\n * Retains only elements of this [MutableIterable] that
match the given [predicate].\n *\n * @return `true` if any element was removed from this collection, or `false` when
all elements were retained and collection was not modified.\n */\npublic fun <T>
MutableIterable<T>.retainAll(predicate: (T) -> Boolean): Boolean = filterInPlace(predicate, false)\n\nprivate fun
<T> MutableIterable<T>.filterInPlace(predicate: (T) -> Boolean, predicateResultToRemove: Boolean): Boolean {\n
var result = false\n with(iterator()) {\n while (hasNext())\n if (predicate(next()) ==
predicateResultToRemove) {\n remove()\n result = true\n }\n }\n return
result\n}\n\n/**\n * Removes the element at the specified [index] from this list.\n * In Kotlin one should use the
[MutableList.removeAt] function instead.\n */\n@Deprecated("Use removeAt(index) instead.",
ReplaceWith("removeAt(index)"), level = DeprecationLevel.ERROR)\n@kotlin.internal.InlineOnly\npublic inline
fun <T> MutableList<T>.remove(index: Int): T = removeAt(index)\n\n/**\n * Removes the first element from this
mutable list and returns that removed element, or throws [NoSuchElementException] if this list is empty.\n
*/\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun <T>
MutableList<T>.removeFirst(): T = if (isEmpty()) throw NoSuchElementException("List is empty.") else
removeAt(0)\n\n/**\n * Removes the first element from this mutable list and returns that removed element, or
returns `null` if this list is empty.\n
*/\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun <T>
MutableList<T>.removeFirstOrNull(): T? = if (isEmpty()) null else removeAt(0)\n\n/**\n * Removes the last
element from this mutable list and returns that removed element, or throws [NoSuchElementException] if this list is
empty.\n */\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun <T>
MutableList<T>.removeLast(): T = if (isEmpty()) throw NoSuchElementException("List is empty.") else
removeAt(lastIndex)\n\n/**\n * Removes the last element from this mutable list and returns that removed element,
or returns `null` if this list is empty.\n
*/\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun <T>
MutableList<T>.removeLastOrNull(): T? = if (isEmpty()) null else removeAt(lastIndex)\n\n/**\n * Removes all
elements from this [MutableList] that match the given [predicate].\n *\n * @return `true` if any element was
removed from this collection, or `false` when no elements were removed and collection was not modified.\n
*/\npublic fun <T> MutableList<T>.removeAll(predicate: (T) -> Boolean): Boolean = filterInPlace(predicate,
true)\n\n/**\n * Retains only elements of this [MutableList] that match the given [predicate].\n *\n * @return `true`
if any element was removed from this collection, or `false` when all elements were retained and collection was not
modified.\n */\npublic fun <T> MutableList<T>.retainAll(predicate: (T) -> Boolean): Boolean =
filterInPlace(predicate, false)\n\nprivate fun <T> MutableList<T>.filterInPlace(predicate: (T) -> Boolean,
predicateResultToRemove: Boolean): Boolean {\n if (this !is RandomAccess)\n return (this as
MutableIterable<T>).filterInPlace(predicate, predicateResultToRemove)\n var writeIndex: Int = 0\n for
(readIndex in 0..lastIndex) {\n val element = this[readIndex]\n if (predicate(element) ==
predicateResultToRemove)\n continue\n\n if (writeIndex != readIndex)\n this[writeIndex] =
element\n writeIndex++\n }\n if (writeIndex < size) {\n for (removeIndex in lastIndex downTo
writeIndex)\n removeAt(removeIndex)\n return true\n } else {\n return false\n }\n}\n\n"/\n\n *
Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is
governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*/\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("CollectionsKt")\n\npackage
kotlin.collections\n\nprivate open class ReversedListReadOnly<out T>(private val delegate: List<T>) :
AbstractList<T>() {\n override val size: Int get() = delegate.size\n override fun get(index: Int): T =
delegate[reverseElementIndex(index)]\n}\n\nprivate class ReversedList<T>(private val delegate: MutableList<T>) :
AbstractMutableList<T>() {\n override val size: Int get() = delegate.size\n override fun get(index: Int): T =
delegate[reverseElementIndex(index)]\n\n override fun clear() = delegate.clear()\n override fun removeAt(index:
Int): T = delegate.removeAt(reverseElementIndex(index))\n\n override fun set(index: Int, element: T): T =

```

```

delegate.set(reverseElementIndex(index), element)\n  override fun add(index: Int, element: T) {\n
delegate.add(reversePositionIndex(index), element)\n  }\n}\n\nprivate fun List<*>.reverseElementIndex(index:
Int) =\n  if (index in 0..lastIndex) lastIndex - index else throw IndexOutOfBoundsException("\u0022Element index
\u0026#x26# $index must be in range [{0..lastIndex}].\u0026#x26#")\n\nprivate fun List<*>.reversePositionIndex(index: Int) =\n  if (index
in 0..size) size - index else throw IndexOutOfBoundsException("\u0022Position index $index must be in range
[{0..size}].\u0026#x26#")\n\n\n/**\n * Returns a reversed read-only view of the original List.\n * All changes made in the
original list will be reflected in the reversed one.\n * @sample samples.collections.ReversedViews.asReversedList\n
*/\n\npublic fun <T> List<T>.asReversed(): List<T> = ReversedListReadOnly(this)\n\n\n/**\n * Returns a reversed
mutable view of the original mutable List.\n * All changes made in the original list will be reflected in the reversed
one and vice versa.\n * @sample samples.collections.ReversedViews.asReversedMutableList\n
*/\n\n@kotlin.jvm.JvmName("\u0022asReversedMutable\u0026#x26#")\n\npublic fun <T> MutableList<T>.asReversed():
MutableList<T> = ReversedList(this)\n\n\n", /*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n
*/\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("\u0022SequencesKt\u0026#x26#")\n@file:OptIn(Experimenta
lTypeInference::class)\n\npackage kotlin.sequences\n\nimport kotlin.coroutines.*\nimport
kotlin.coroutines.intrinsics.*\nimport kotlin.experimental.ExperimentalTypeInference\n\n\n/**\n * Builds a
[Sequence] lazily yielding values one by one.\n * @see kotlin.sequences.generateSequence\n * @sample
samples.collections.Sequences.Building.buildSequenceYieldAll\n * @sample
samples.collections.Sequences.Building.buildFibonacciSequence\n
*/\n\n@SinceKotlin("1.3")\n\npublic fun <T>
sequence(@BuilderInference block: suspend SequenceScope<T>().() -> Unit): Sequence<T> = Sequence {
iterator(block) }\n\n\n@SinceKotlin("1.3")\n\n@Deprecated("\u0022Use 'sequence { }' function instead.\u0026#x26#",
ReplaceWith("\u0022sequence(builderAction)\u0026#x26#"), level =
DeprecationLevel.ERROR)\n\n@kotlin.internal.InlineOnly\n\npublic inline fun <T> buildSequence(@BuilderInference
noinline builderAction: suspend SequenceScope<T>().() -> Unit): Sequence<T> = Sequence { iterator(builderAction)
}\n\n\n/**\n * Builds an [Iterator] lazily yielding values one by one.\n * @sample
samples.collections.Sequences.Building.buildIterator\n * @sample samples.collections.Iterables.Building.iterable\n
*/\n\n@SinceKotlin("1.3")\n\npublic fun <T> iterator(@BuilderInference block: suspend SequenceScope<T>().() ->
Unit): Iterator<T> {\n  val iterator = SequenceBuilderIterator<T>()\n  iterator.nextStep =
block.createCoroutineUnintercepted(receiver = iterator, completion = iterator)\n  return
iterator\n}\n\n\n@SinceKotlin("1.3")\n\n@Deprecated("\u0022Use 'iterator { }' function instead.\u0026#x26#",
ReplaceWith("\u0022iterator(builderAction)\u0026#x26#"), level = DeprecationLevel.ERROR)\n\n@kotlin.internal.InlineOnly\n\npublic
inline fun <T> buildIterator(@BuilderInference noinline builderAction: suspend SequenceScope<T>().() -> Unit):
Iterator<T> = iterator(builderAction)\n\n\n/**\n * The scope for yielding values of a [Sequence] or an [Iterator],
provides [yield] and [yieldAll] suspension functions.\n * @see sequence\n * @see iterator\n * @sample
samples.collections.Sequences.Building.buildSequenceYieldAll\n * @sample
samples.collections.Sequences.Building.buildFibonacciSequence\n
*/\n\n@RestrictsSuspension\n@SinceKotlin("1.3")\n\npublic abstract class SequenceScope<in T> internal
constructor() {\n  /**\n   * Yields a value to the [Iterator] being built and suspends\n   * until the next value is
requested.\n   * @sample samples.collections.Sequences.Building.buildSequenceYieldAll\n   * @sample
samples.collections.Sequences.Building.buildFibonacciSequence\n   */\n   public abstract suspend fun yield(value:
T)\n\n   /**\n   * Yields all values from the `iterator` to the [Iterator] being built\n   * and suspends until all these
values are iterated and the next one is requested.\n   * @sample samples.collections.Sequences.Building.buildSequenceYieldAll\n
*/\n   public abstract suspend fun yieldAll(iterator: Iterator<T>)\n\n   /**\n   * Yields a collections of values to
the [Iterator] being built\n   * and suspends until all these values are iterated and the next one is requested.\n   *
@sample samples.collections.Sequences.Building.buildSequenceYieldAll\n   */\n   public suspend fun
yieldAll(elements: Iterable<T>) {\n     if (elements is Collection && elements.isEmpty()) return\n     return

```





```

    while (array[j] > pivot)\n        j--\n        if (i <= j) {\n            val tmp = array[i]\n            array[i] = array[j]\n            array[j] = tmp\n            i++\n            j--\n        }\n    }\n    return i\n}\n\n@ExperimentalUnsignedTypes\nprivate fun quickSort(\n    array: UByteArray, left: Int, right: Int) {\n    val index = partition(array, left, right)\n    if (left < index - 1)\n        quickSort(array, left, index - 1)\n    if (index < right)\n        quickSort(array, index, right)\n}\n\n//\nUShortArray\n\n=====\n@Exp\nperimentalUnsignedTypes\nprivate fun partition(\n    array: UShortArray, left: Int, right: Int): Int {\n    var i = left\n    var j = right\n    val pivot = array[(left + right) / 2]\n    while (i <= j) {\n        while (array[i] < pivot)\n            i++\n        while (array[j] > pivot)\n            j--\n        if (i <= j) {\n            val tmp = array[i]\n            array[i] = array[j]\n            array[j] = tmp\n            i++\n            j--\n        }\n    }\n    return i\n}\n\n@ExperimentalUnsignedTypes\nprivate fun quickSort(\n    array: UShortArray, left: Int, right: Int) {\n    val index = partition(array, left, right)\n    if (left < index - 1)\n        quickSort(array, left, index - 1)\n    if (index < right)\n        quickSort(array, index, right)\n}\n\n//\nUIntArray\n\n=====\n@Exp\nperimentalUnsignedTypes\nprivate fun partition(\n    array: UIntArray, left: Int, right: Int): Int {\n    var i = left\n    var j = right\n    val pivot = array[(left + right) / 2]\n    while (i <= j) {\n        while (array[i] < pivot)\n            i++\n        while (array[j] > pivot)\n            j--\n        if (i <= j) {\n            val tmp = array[i]\n            array[i] = array[j]\n            array[j] = tmp\n            i++\n            j--\n        }\n    }\n    return i\n}\n\n@ExperimentalUnsignedTypes\nprivate fun quickSort(\n    array: UIntArray, left: Int, right: Int) {\n    val index = partition(array, left, right)\n    if (left < index - 1)\n        quickSort(array, left, index - 1)\n    if (index < right)\n        quickSort(array, index, right)\n}\n\n//\nULongArray\n\n=====\n@Exp\nperimentalUnsignedTypes\nprivate fun partition(\n    array: ULongArray, left: Int, right: Int): Int {\n    var i = left\n    var j = right\n    val pivot = array[(left + right) / 2]\n    while (i <= j) {\n        while (array[i] < pivot)\n            i++\n        while (array[j] > pivot)\n            j--\n        if (i <= j) {\n            val tmp = array[i]\n            array[i] = array[j]\n            array[j] = tmp\n            i++\n            j--\n        }\n    }\n    return i\n}\n\n@ExperimentalUnsignedTypes\nprivate fun quickSort(\n    array: ULongArray, left: Int, right: Int) {\n    val index = partition(array, left, right)\n    if (left < index - 1)\n        quickSort(array, left, index - 1)\n    if (index < right)\n        quickSort(array, index, right)\n}\n\n//\nInterfaces\n\n=====\n\n/*\n\n* Sorts the given array using qsort algorithm.\n */\n\n@ExperimentalUnsignedTypes\ninternal fun sortArray(array: UByteArray, fromIndex: Int, toIndex: Int) = quickSort(array, fromIndex, toIndex - 1)\n\n@ExperimentalUnsignedTypes\ninternal fun sortArray(array: UShortArray, fromIndex: Int, toIndex: Int) = quickSort(array, fromIndex, toIndex - 1)\n\n@ExperimentalUnsignedTypes\ninternal fun sortArray(array: UIntArray, fromIndex: Int, toIndex: Int) = quickSort(array, fromIndex, toIndex - 1)\n\n@ExperimentalUnsignedTypes\ninternal fun sortArray(array: ULongArray, fromIndex: Int, toIndex: Int) = quickSort(array, fromIndex, toIndex - 1)\n\n/*\n\n* Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin\n\nimport kotlin.internal.InlineOnly\n\n/**\n * Compares this object with the specified object for order. Returns zero if this object is equal\n * to the specified [other] object, a negative number if it's less than [other], or a positive number\n * if it's greater than [other].\n */\n * This function delegates to [Comparable.compareTo] and allows to call it in infix form.\n\n\n@InlineOnly\n@SinceKotlin("1.6")\npublic inline infix fun <T> Comparable<T>.compareTo(other: T): Int =\n    this.compareTo(other)\n\n/*\n\n* Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.contracts\n\nimport kotlin.internal.ContractsDsl\nimport kotlin.internal.InlineOnly\n\n/**\n * This marker distinguishes the experimental contract declaration API and is used to opt-in for that feature\n * when declaring contracts of user functions.\n */\n * Any usage of a declaration

```

annotated with `@ExperimentalContracts` must be accepted either by `* annotating that usage with the [OptIn] annotation, e.g. @OptIn(ExperimentalContracts::class), * or by using the compiler argument -Xopt-in=kotlin.contracts.ExperimentalContracts.`

```

*/\n@Suppress("DEPRECATION")\n@Retention(AnnotationRetention.BINARY)\n@SinceKotlin("1.3")\n@Experimental\n@RequiresOptIn\n@MustBeDocumented\npublic annotation class ExperimentalContracts\n\n/**\n * Provides a scope, where the functions of the contract DSL, such as [returns], [callsInPlace], etc.,\n * can be used to describe the contract of a function.\n * This type is used as a receiver type of the lambda function passed to the [contract] function.\n * \n * @see contract\n */\n\n*/\n@ContractsDsl\n@ExperimentalContracts\n@SinceKotlin("1.3")\npublic interface ContractBuilder {\n    /**\n     * Describes a situation when a function returns normally, without any exceptions thrown.\n     * \n     * Use [SimpleEffect.implies] function to describe a conditional effect that happens in such case.\n     * \n     * \n     * // @sample samples.contracts.returnsContract\n     * @ContractsDsl public fun returns(): Returns\n     * \n     * Describes a situation when a function returns normally with the specified return [value].\n     * \n     * The possible values of [value] are limited to `true`, `false` or `null`.\n     * \n     * Use [SimpleEffect.implies] function to describe a conditional effect that happens in such case.\n     * \n     * // @sample samples.contracts.returnsTrueContract\n     * // @sample samples.contracts.returnsFalseContract\n     * // @sample samples.contracts.returnsNullContract\n     * @ContractsDsl public fun returns(value: Any?): Returns\n     * \n     * // @sample\n     * // @sample\n     * Describes a situation when a function returns normally with any value that is not `null`.\n     * \n     * Use [SimpleEffect.implies] function to describe a conditional effect that happens in such case.\n     * \n     * // @sample\n     * samples.contracts.returnsNotNullContract\n     * @ContractsDsl public fun returnsNotNull(): ReturnsNotNull\n     * \n     * // @sample\n     * Specifies that the function parameter [lambda] is invoked in place.\n     * \n     * This contract specifies that:\n     * \n     * 1. the function [lambda] can only be invoked during the call of the owner function,\n     * \n     * and it won't be invoked after that owner function call is completed;\n     * \n     * 2. _(optionally)_ the function [lambda] is invoked the amount of times specified by the [kind] parameter.\n     * \n     * see the [InvocationKind] enum for possible values.\n     * \n     * A function declaring the `callsInPlace` effect must be _inline_.\n     * \n     * \n     * // @sample\n     * samples.contracts.callsInPlaceAtMostOnceContract\n     * \n     * @sample\n     * samples.contracts.callsInPlaceAtLeastOnceContract\n     * \n     * @sample\n     * samples.contracts.callsInPlaceExactlyOnceContract\n     * \n     * @sample\n     * samples.contracts.callsInPlaceUnknownContract\n     * \n     * // @sample\n     * @ContractsDsl public fun <R> callsInPlace(lambda: Function<R>, kind: InvocationKind = InvocationKind.UNKNOWN): CallsInPlace\n     * \n     * // @sample\n     * Specifies how many times a function invokes its function parameter in place.\n     * \n     * See [ContractBuilder.callsInPlace] for the details of the call-in-place function contract.\n     * \n     * \n     * // @sample\n     * samples.contracts.callsInPlaceAtMostOnceContract\n     * \n     * @ContractsDsl AT_MOST_ONCE,\n     * \n     * // @sample\n     * A function parameter will be invoked one time or not invoked at all.\n     * \n     * \n     * // @sample\n     * samples.contracts.callsInPlaceAtLeastOnceContract\n     * \n     * @ContractsDsl AT_LEAST_ONCE,\n     * \n     * // @sample\n     * A function parameter will be invoked one or more times.\n     * \n     * \n     * // @sample\n     * samples.contracts.callsInPlaceAtLeastOnceContract\n     * \n     * @ContractsDsl AT_LEAST_ONCE,\n     * \n     * // @sample\n     * A function parameter will be invoked exactly one time.\n     * \n     * \n     * // @sample\n     * samples.contracts.callsInPlaceExactlyOnceContract\n     * \n     * @ContractsDsl EXACTLY_ONCE,\n     * \n     * // @sample\n     * A function parameter is called in place, but it's unknown how many times it can be called.\n     * \n     * \n     * // @sample\n     * samples.contracts.callsInPlaceUnknownContract\n     * \n     * @ContractsDsl UNKNOWN\n     * \n     * // @sample\n     * Specifies the contract of a function.\n     * \n     * The contract description must be at the beginning of a function and have at least one effect.\n     * \n     * \n     * Only the top-level functions can have a contract for now.\n     * \n     * \n     * @param builder the lambda where the contract of a function is described with the help of the [ContractBuilder] members.\n     * \n     * \n     * // @sample\n     * samples.contracts.returnsContract\n     * \n     * @sample\n     * samples.contracts.returnsTrueContract\n     * \n     * @sample\n     * samples.contracts.returnsFalseContract\n     * \n     * @sample\n     * samples.contracts.returnsNullContract\n     * \n     * @sample\n     * samples.contracts.returnsNotNullContract\n     * \n     * @sample\n     * samples.contracts.callsInPlaceAtMostOnceContract\n     * \n     * @sample\n     * samples.contracts.callsInPlaceAtLeastOnceContract\n     * \n     * @sample

```

```

samples.contracts.callsInPlaceExactlyOnceContract\n* @sample
samples.contracts.callsInPlaceUnknownContract\n*\n@ContractsDsl\n@ExperimentalContracts\n@InlineOnly\n@
SinceKotlin("1.3")\n@Suppress("UNUSED_PARAMETER")\npublic inline fun contract(builder:
ContractBuilder.() -> Unit) { }\n", "\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n *\n\npackage kotlin.coroutines\n\n/**\n * Marks coroutine context element that
intercepts coroutine continuations.\n * The coroutines framework uses [ContinuationInterceptor.Key] to retrieve the
interceptor and\n * intercepts all coroutine continuations with [interceptContinuation] invocations.\n *\n *
[ContinuationInterceptor] behaves like a [polymorphic element][AbstractCoroutineContextKey], meaning that\n *
its implementation delegates [get][CoroutineContext.Element.get] and
[minusKey][CoroutineContext.Element.minusKey]\n * to [getPolymorphicElement] and [minusPolymorphicKey]
respectively.\n * [ContinuationInterceptor] subtypes can be extracted from the coroutine context using either
[ContinuationInterceptor.Key]\n * or subtype key if it extends [AbstractCoroutineContextKey].\n
*\n\n@SinceKotlin("1.3")\npublic interface ContinuationInterceptor : CoroutineContext.Element {\n /**\n *
The key that defines *the* context interceptor.\n */\n companion object Key :
CoroutineContext.Key<ContinuationInterceptor>\n\n /**\n * Returns continuation that wraps the original
[continuation], thus intercepting all resumptions.\n * This function is invoked by coroutines framework when
needed and the resulting continuations are\n * cached internally per each instance of the original [continuation].\n
*\n * This function may simply return original [continuation] if it does not want to intercept this particular
continuation.\n *\n * When the original [continuation] completes, coroutine framework invokes
[releaseInterceptedContinuation]\n * with the resulting continuation if it was intercepted, that is if
`interceptContinuation` had previously\n * returned a different continuation instance.\n */\n public fun <T>
interceptContinuation(continuation: Continuation<T>): Continuation<T>\n\n /**\n * Invoked for the
continuation instance returned by [interceptContinuation] when the original\n * continuation completes and will
not be used anymore. This function is invoked only if [interceptContinuation]\n * had returned a different
continuation instance from the one it was invoked with.\n *\n * Default implementation does nothing.\n */\n
*\n * @param continuation Continuation instance returned by this interceptor's [interceptContinuation] invocation.\n
*\n\n public fun releaseInterceptedContinuation(continuation: Continuation<*>) {\n /* do nothing by default
*\n */\n }\n\n public override operator fun <E : CoroutineContext.Element> get(key: CoroutineContext.Key<E>):
E? {\n // getPolymorphicKey specialized for ContinuationInterceptor key\n
*\n\n @OptIn(ExperimentalStdlibApi::class)\n if (key is AbstractCoroutineContextKey<*, *>) {\n
*\n\n @Suppress("UNCHECKED_CAST")\n return if (key.isSubKey(this.key)) key.tryCast(this) as? E else
null\n }\n\n @Suppress("UNCHECKED_CAST")\n return if (ContinuationInterceptor === key) this as
E else null\n }\n\n\n public override fun minusKey(key: CoroutineContext.Key<*>): CoroutineContext {\n
*\n\n // minusPolymorphicKey specialized for ContinuationInterceptor key\n
*\n\n @OptIn(ExperimentalStdlibApi::class)\n if (key is AbstractCoroutineContextKey<*, *>) {\n return if
(key.isSubKey(this.key) && key.tryCast(this) != null) EmptyCoroutineContext else this\n }\n return if
(ContinuationInterceptor === key) EmptyCoroutineContext else this\n }\n\n", "\n * Copyright 2010-2018
JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n *\n\npackage kotlin.coroutines\n\n/**\n *
Persistent context for the coroutine. It is an indexed set of [Element] instances.\n * An indexed set is a mix between
a set and a map.\n * Every element in this set has a unique [Key].\n */\n\n@SinceKotlin("1.3")\npublic interface
CoroutineContext {\n /**\n * Returns the element with the given [key] from this context or `null`.\n */\n
*\n\n public operator fun <E : Element> get(key: Key<E>): E?\n\n /**\n * Accumulates entries of this context
starting with [initial] value and applying [operation]\n * from left to right to current accumulator value and each
element of this context.\n */\n\n *\n\n public fun <R> fold(initial: R, operation: (R, Element) -> R): R\n\n /**\n *
Returns a context containing elements from this context and elements from other [context].\n * The elements
from this context with the same key as in the other one are dropped.\n */\n\n *\n\n public operator fun plus(context:

```

```

CoroutineContext): CoroutineContext =\n    if (context === EmptyCoroutineContext) this else // fast path -- avoid
lambda creation\n    context.fold(this) { acc, element ->\n        val removed =
acc.minusKey(element.key)\n        if (removed === EmptyCoroutineContext) element else {\n            //
make sure interceptor is always last in the context (and thus is fast to get when present)\n            val interceptor
= removed[ContinuationInterceptor]\n            if (interceptor == null) CombinedContext(removed, element) else
{\n                val left = removed.minusKey(ContinuationInterceptor)\n                if (left ===
EmptyCoroutineContext) CombinedContext(element, interceptor) else\n                CombinedContext(CombinedContext(left, element), interceptor)\n            }\n        }\n    }\n\n    /**\n    * Returns a context containing elements from this context, but without an element with\n    * the specified [key].\n    */\n    public fun minusKey(key: Key<*>): CoroutineContext\n\n    /**\n    * Key for the elements of
[CoroutineContext]. [E] is a type of element with this key.\n    */\n    public interface Key<E : Element>\n\n    /**\n    * An element of the [CoroutineContext]. An element of the coroutine context is a singleton context by itself.\n    */\n    public interface Element : CoroutineContext {\n\n        /**\n        * A key of this coroutine context element.\n        */\n        public val key: Key<*>\n\n        public override operator fun <E : Element> get(key: Key<E>): E? =\n        @Suppress("UNCHECKED_CAST")\n        if (this.key == key) this as E else null\n\n        public override
fun <R> fold(initial: R, operation: (R, Element) -> R): R =\n        operation(initial, this)\n\n        public override
fun minusKey(key: Key<*>): CoroutineContext =\n        if (this.key == key) EmptyCoroutineContext else this\n    }\n\n    "/*\n    * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n    * Use of
this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n    */\n\n    \npackage kotlin.coroutines\n\nimport kotlin.coroutines.CoroutineContext.Element\nimport
kotlin.coroutines.CoroutineContext.Key\n\n/**\n    * Base class for [CoroutineContext.Element] implementations.\n    */\n    @SinceKotlin("1.3")\n    public abstract class AbstractCoroutineContextElement(public override val key:
Key<*>) : Element\n\n    /**\n    * Base class for [CoroutineContext.Key] associated with polymorphic
[CoroutineContext.Element] implementation.\n    * Polymorphic element implementation implies delegating its
[get][Element.get] and [minusKey][Element.minusKey]\n    * to [getPolymorphicElement] and
[minusPolymorphicKey] respectively.\n    * Polymorphic elements can be extracted from the coroutine context
using both element key and its supertype key.\n    * Example of polymorphic elements:\n    * ```\n    * open class
BaseElement : CoroutineContext.Element {\n    *     companion object Key : CoroutineContext.Key<BaseElement>\n    *
    *     override val key: CoroutineContext.Key<*> get() = Key\n    *     // It is important to use getPolymorphicKey and
minusPolymorphicKey\n    *     override fun <E : CoroutineContext.Element> get(key: CoroutineContext.Key<E>):
E? = getPolymorphicElement(key)\n    *     override fun minusKey(key: CoroutineContext.Key<*>):
CoroutineContext = minusPolymorphicKey(key)\n    * }\n    * class DerivedElement : BaseElement() {\n    *
    *     companion object Key : AbstractCoroutineContextKey<BaseElement, DerivedElement>(BaseElement, { it as?
DerivedElement })\n    * }\n    * // Now it is possible to query both `BaseElement` and `DerivedElement`\n    *
someContext[BaseElement] // Returns BaseElement?, non-null both for BaseElement and DerivedElement
instances\n    * someContext[DerivedElement] // Returns DerivedElement?, non-null only for DerivedElement
instance\n    * ```\n    * @param B base class of a polymorphic element\n    * @param baseKey an instance of base key\n    *
    * @param E element type associated with the current key\n    * @param safeCast a function that can safely cast
abstract [CoroutineContext.Element] to the concrete [E] type\n    *         and return the element if it is a subtype
of [E] or `null` otherwise.\n    */\n    @SinceKotlin("1.3")\n    @ExperimentalStdlibApi\n    public abstract class
AbstractCoroutineContextKey<B : Element, E : B>(\n        baseKey: Key<B>,\n        private val safeCast: (element:
Element) -> E?) : Key<E> {\n        private val topmostKey: Key<*> = if (baseKey is
AbstractCoroutineContextKey<*, *>) baseKey.topmostKey else baseKey\n\n        internal fun tryCast(element:
Element): E? = safeCast(element)\n        internal fun isSubKey(key: Key<*>): Boolean = key === this || topmostKey
=== key\n    }\n\n    /**\n    * Returns the current element if it is associated with the given [key] in a polymorphic manner
or `null` otherwise.\n    * This method returns non-null value if either [Element.key] is equal to the given [key] or if
the [key] is associated\n    * with [Element.key] via [AbstractCoroutineContextKey].\n    * See
[AbstractCoroutineContextKey] for the example of usage.\n    */

```

```

*\n@SinceKotlin("1.3")\n@ExperimentalStdlibApi\npublic fun <E : Element>
Element.getPolymorphicElement(key: Key<E>): E? {\n  if (key is AbstractCoroutineContextKey<*, *>) {\n
@Suppress("UNCHECKED_CAST")\n    return if (key.isSubKey(this.key)) key.tryCast(this) as? E else null\n
}\n  @Suppress("UNCHECKED_CAST")\n    return if (this.key === key) this as E else null\n}\n\n/**\n *
Returns empty coroutine context if the element is associated with the given [key] in a polymorphic manner\n * or
`null` otherwise.\n * This method returns empty context if either [Element.key] is equal to the given [key] or if the
[key] is associated\n * with [Element.key] via [AbstractCoroutineContextKey].\n * See
[AbstractCoroutineContextKey] for the example of usage.\n
*\n@SinceKotlin("1.3")\n@ExperimentalStdlibApi\npublic fun Element.minusPolymorphicKey(key: Key<*>):
CoroutineContext {\n  if (key is AbstractCoroutineContextKey<*, *>) {\n    return if (key.isSubKey(this.key)
&& key.tryCast(this) != null) EmptyCoroutineContext else this\n  }\n  return if (this.key === key)
EmptyCoroutineContext else this\n}\n\n/**\n * An empty coroutine context.\n */\n@SinceKotlin("1.3")\npublic
object EmptyCoroutineContext : CoroutineContext, Serializable {\n  private const val serialVersionUID: Long =
0\n  private fun readResolve(): Any = EmptyCoroutineContext\n\n  public override fun <E : Element> get(key:
Key<E>): E? = null\n  public override fun <R> fold(initial: R, operation: (R, Element) -> R): R = initial\n
public override fun plus(context: CoroutineContext): CoroutineContext = context\n  public override fun
minusKey(key: Key<*>): CoroutineContext = this\n  public override fun hashCode(): Int = 0\n  public override
fun toString(): String = "EmptyCoroutineContext"\n}\n\n//----- internal impl -----\n\n// this class is not
exposed, but is hidden inside implementations\n// this is a left-biased list, so that `plus` works
naturally\n@SinceKotlin("1.3")\ninternal class CombinedContext(\n  private val left: CoroutineContext,\n
private val element: Element\n) : CoroutineContext, Serializable {\n\n  override fun <E : Element> get(key:
Key<E>): E? {\n    var cur = this\n    while (true) {\n      cur.element[key]?.let { return it }\n      val next
= cur.left\n      if (next is CombinedContext) {\n        cur = next\n      } else {\n        return
next[key]\n      }\n    }\n\n    public override fun <R> fold(initial: R, operation: (R, Element) -> R): R =\n
operation(left.fold(initial, operation), element)\n\n    public override fun minusKey(key: Key<*>):
CoroutineContext {\n      element[key]?.let { return left }\n      val newLeft = left.minusKey(key)\n      return
when {\n        newLeft === left -> this\n        newLeft === EmptyCoroutineContext -> element\n      } else ->
CombinedContext(newLeft, element)\n    }\n\n    private fun size(): Int {\n      var cur = this\n      var size
= 2\n      while (true) {\n        cur = cur.left as? CombinedContext ?: return size\n        size++\n      }\n    }\n\n
private fun contains(element: Element): Boolean =\n      get(element.key) == element\n\n    private fun
containsAll(context: CombinedContext): Boolean {\n      var cur = context\n      while (true) {\n        if
(!contains(cur.element)) return false\n        val next = cur.left\n        if (next is CombinedContext) {\n
cur = next\n        } else {\n          return contains(next as Element)\n        }\n      }\n    }\n\n    override fun
equals(other: Any?): Boolean =\n      this === other || other is CombinedContext && other.size() == size() &&
other.containsAll(this)\n\n    override fun hashCode(): Int = left.hashCode() + element.hashCode()\n\n    override
fun toString(): String = "\n    "[" + fold("") { acc, element ->\n      if (acc.isEmpty()) element.toString() else
"$acc, $element"\n    } + "]" + "\n\n    private fun writeReplace(): Any {\n      val n = size()\n      val elements =
arrayOfNulls<CoroutineContext>(n)\n      var index = 0\n      fold(Unit) { _, element -> elements[index++] =
element }\n      check(index == n)\n      @Suppress("UNCHECKED_CAST")\n      return Serialized(elements
as Array<CoroutineContext>)\n    }\n\n    private class Serialized(val elements: Array<CoroutineContext>) :
Serializable {\n      companion object {\n        private const val serialVersionUID: Long = 0L\n      }\n\n      private fun readResolve(): Any = elements.fold(EmptyCoroutineContext, CoroutineContext::plus)\n    }\n}\n\n"/*\n *
Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code
is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*\n\n@file:kotlin.jvm.JvmName("IntrinsicsKt")\n@file:kotlin.jvm.JvmMultifileClass\n\npackage
kotlin.coroutines.intrinsics\n\nimport kotlin.contracts.*\nimport kotlin.coroutines.*\nimport
kotlin.internal.InlineOnly\n\n/**\n * Obtains the current continuation instance inside suspend functions and either
suspends\n * currently running coroutine or returns result immediately without suspension.\n * If the [block]

```

returns the special [COROUTINE\_SUSPENDED] value, it means that suspend function did suspend the execution and will not return any result immediately. In this case, the [Continuation] provided to the [block] shall be resumed by invoking [Continuation.resumeWith] at some moment in the future when the result becomes available to resume the computation. Otherwise, the return value of the [block] must have a type assignable to [T] and represents the result of this suspend function. It means that the execution was not suspended and the [Continuation] provided to the [block] shall not be invoked. As the result type of the [block] is declared as `Any?` and cannot be correctly type-checked, its proper return type remains on the conscience of the suspend function's author. Invocation of [Continuation.resumeWith] resumes coroutine directly in the invoker's thread without going through the [ContinuationInterceptor] that might be present in the coroutine's [CoroutineContext]. It is the invoker's responsibility to ensure that a proper invocation context is established. [Continuation.intercepted] can be used to acquire the intercepted continuation. Note that it is not recommended to call either [Continuation.resume] nor [Continuation.resumeWithException] functions synchronously in the same stackframe where suspension function is run. Use [suspendCoroutine] as a safer way to obtain current continuation instance.

```
*\n@SinceKotlin("1.3")\n@InlineOnly\n@Suppress("UNUSED_PARAMETER",
"RedundantSuspendModifier")\npublic suspend inline fun <T>
suspendCoroutineUninterceptedOrReturn(crossinline block: (Continuation<T>) -> Any?): T {\n  contract {
callsInPlace(block, InvocationKind.EXACTLY_ONCE) }\n  throw NotImplementedError("Implementation of
suspendCoroutineUninterceptedOrReturn is intrinsic")\n}\n\n/*\n * This value is used as a return value of
[suspendCoroutineUninterceptedOrReturn] `block` argument to state that\n * the execution was suspended and will
not return any result immediately.\n * **Note: this value should not be used in general code.** Using it outside
of the context of\n * `suspendCoroutineUninterceptedOrReturn` function return value (including, but not limited
to,\n * storing this value in other properties, returning it from other functions, etc)\n * can lead to unspecified
behavior of the code.\n */\n// It is implemented as property with getter to avoid ProGuard <clinit> problem with
multifile IntrinsicsKt class\n@SinceKotlin("1.3")\npublic val COROUTINE_SUSPENDED: Any get() =
CoroutineSingletons.COROUTINE_SUSPENDED\n\n// Using enum here ensures two important properties:\n// 1.
It makes SafeContinuation serializable with all kinds of serialization frameworks (since all of them natively support
enums)\n// 2. It improves debugging experience, since you clearly see toString() value of those objects and what
package they come from\n@SinceKotlin("1.3")\n@PublishedApi // This class is Published API via serialized
representation of SafeContinuation, don't rename/move\ninternal enum class CoroutineSingletons {\n
COROUTINE_SUSPENDED, UNDECIDED, RESUMED }\n\n/*\n * Copyright 2010-2018 JetBrains s.r.o. and
Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.experimental\n\n/** Performs a bitwise AND
operation between the two values. *\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline infix fun
Byte.and(other: Byte): Byte = (this.toInt() and other.toInt()).toByte()\n\n/** Performs a bitwise OR operation
between the two values. *\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline infix fun
Byte.or(other: Byte): Byte = (this.toInt() or other.toInt()).toByte()\n\n/** Performs a bitwise XOR operation
between the two values. *\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline infix fun
Byte.xor(other: Byte): Byte = (this.toInt() xor other.toInt()).toByte()\n\n/** Inverts the bits in this value.
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline fun Byte.inv(): Byte =
(this.toInt().inv()).toByte()\n\n/** Performs a bitwise AND operation between the two values.
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline infix fun Short.and(other: Short): Short =
(this.toInt() and other.toInt()).toShort()\n\n/** Performs a bitwise OR operation between the two values.
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline infix fun Short.or(other: Short): Short =
(this.toInt() or other.toInt()).toShort()\n\n/** Performs a bitwise XOR operation between the two values.
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline infix fun Short.xor(other: Short): Short =
(this.toInt() xor other.toInt()).toShort()\n\n/** Inverts the bits in this value.
*\n@SinceKotlin("1.1")\n@kotlin.internal.InlineOnly\npublic inline fun Short.inv(): Short =
```

(this.toInt().inv()).toShort()\n\n", "/\*\n \* Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n \* Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n \*/\n\npackage kotlin.experimental\n\n/\*\*\n \* The experimental marker for type inference augmenting annotations.\n \* Any usage of a declaration annotated with `@ExperimentalTypeInference` must be accepted either by\n \* annotating that usage with the [OptIn] annotation, e.g. `@OptIn(ExperimentalTypeInference::class)`,\n \* or by using the compiler argument `-Xopt-in=kotlin.experimental.ExperimentalTypeInference`.\n \* @Suppress("DEPRECATION")\n \* @Experimental(level = Experimental.Level.ERROR)\n \* @RequiresOptIn(level = RequiresOptIn.Level.ERROR)\n \* @MustBeDocumented\n \* @Retention(AnnotationRetention.BINARY)\n \* @Target(AnnotationTarget.ANNOTATION\_CLASS)\n \* @SinceKotlin("1.3")\n \*/\npublic annotation class ExperimentalTypeInference\n", "/\*\n \* Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n \* Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n \*/\n\npackage kotlin.internal\n\n/\*\*\n \* Specifies that the corresponding type should be ignored during type inference.\n \*\n \* @Target(AnnotationTarget.TYPE)\n \* @Retention(AnnotationRetention.BINARY)\n \*/\ninternal annotation class NoInfer\n\n/\*\*\n \* Specifies that the constraint built for the type during type inference should be an equality one.\n \*\n \* @Target(AnnotationTarget.TYPE)\n \* @Retention(AnnotationRetention.BINARY)\n \*/\ninternal annotation class Exact\n\n/\*\*\n \* Specifies that a corresponding member has the lowest priority in overload resolution.\n \*\n \* @Target(AnnotationTarget.FUNCTION, AnnotationTarget.PROPERTY)\n \* @Retention(AnnotationRetention.BINARY)\n \*/\ninternal annotation class LowPriorityInOverloadResolution\n\n/\*\*\n \* Specifies that the corresponding member has the highest priority in overload resolution. Effectively this means that\n \* an extension annotated with this annotation will win in overload resolution over a member with the same signature.\n \*\n \* @Target(AnnotationTarget.FUNCTION, AnnotationTarget.PROPERTY)\n \* @Retention(AnnotationRetention.BINARY)\n \*/\ninternal annotation class HidesMembers\n\n/\*\*\n \* The value of this type parameter should be mentioned in input types (argument types, receiver type or expected type).\n \*\n \* @Target(AnnotationTarget.TYPE\_PARAMETER)\n \* @Retention(AnnotationRetention.BINARY)\n \*/\ninternal annotation class OnlyInputTypes\n\n/\*\*\n \* Specifies that this function should not be called directly without inlining\n \*\n \* @Target(AnnotationTarget.FUNCTION, AnnotationTarget.PROPERTY, AnnotationTarget.PROPERTY\_GETTER, AnnotationTarget.PROPERTY\_SETTER)\n \* @Retention(AnnotationRetention.BINARY)\n \*/\ninternal annotation class InlineOnly\n\n/\*\*\n \* Specifies that this declaration can have dynamic receiver type.\n \*\n \* @Target(AnnotationTarget.FUNCTION, AnnotationTarget.PROPERTY)\n \* @Retention(AnnotationRetention.BINARY)\n \*/\ninternal annotation class DynamicExtension\n\n/\*\*\n \* The value of this parameter should be a property reference expression (`this::foo`), referencing a `lateinit` property,\n \* the backing field of which is accessible at the point where the corresponding argument is passed.\n \*\n \* @Target(AnnotationTarget.VALUE\_PARAMETER)\n \* @Retention(AnnotationRetention.BINARY)\n \* @SinceKotlin("1.2")\n \*/\ninternal annotation class AccessibleLateinitPropertyLiteral\n\n/\*\*\n \* Specifies that this declaration is only completely supported since the specified version.\n \* The Kotlin compiler of an earlier version is going to report a diagnostic on usages of this declaration.\n \* The diagnostic message can be specified with [message], or via [errorCode] (takes less space, but might not be immediately clear\n \* to the user). The diagnostic severity can be specified with [level]: WARNING/ERROR mean that either a warning or an error\n \* is going to be reported, HIDDEN means that the declaration is going to be removed from resolution completely.\n \* [versionKind] specifies which version should be compared with the [version] value, when compiling the usage of the annotated declaration.\n \* Note that prior to 1.2, only [RequireKotlinVersionKind.LANGUAGE\_VERSION] was supported, so the Kotlin compiler before 1.2 is going to\n \* treat any [RequireKotlin] as if it requires the language version. Since 1.2, the Kotlin compiler supports\n \* [RequireKotlinVersionKind.LANGUAGE\_VERSION],

```

[RequireKotlinVersionKind.COMPILER_VERSION] and [RequireKotlinVersionKind.API_VERSION].\n * If the
actual value of [versionKind] is something different (e.g. a new version kind, added in future versions of Kotlin),\n *
Kotlin 1.2 is going to ignore this [RequireKotlin] altogether, where as Kotlin before 1.2 is going to treat this as a
requirement\n * on the language version.\n *\n * This annotation is erased at compile time; its arguments are stored
in a more compact form in the Kotlin metadata.\n *\n @Target(AnnotationTarget.CLASS,
AnnotationTarget.FUNCTION, AnnotationTarget.PROPERTY, AnnotationTarget.CONSTRUCTOR,
AnnotationTarget.TYPEALIAS)\n @Retention(AnnotationRetention.SOURCE)\n @Repeatable\n @SinceKotlin("1.
2")\n internal annotation class RequireKotlin(\n val version: String,\n val message: String = "",\n val level:
DeprecationLevel = DeprecationLevel.ERROR,\n val versionKind: RequireKotlinVersionKind =
RequireKotlinVersionKind.LANGUAGE_VERSION,\n val errorCode: Int = -1)\n\n /**\n * The kind of the
version that is required by [RequireKotlin].\n *\n @SinceKotlin("1.2")\n internal enum class
RequireKotlinVersionKind {\n LANGUAGE_VERSION,\n COMPILER_VERSION,\n
API_VERSION,\n}\n\n /**\n * Specifies that this declaration is a part of special DSL, used for constructing
function's contract.\n *\n @Retention(AnnotationRetention.BINARY)\n @SinceKotlin("1.2")\n internal annotation
class ContractsDsl\n", "\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n *\n\n package kotlin.properties\n\n import kotlin.reflect.KProperty\n\n /**\n * Standard
property delegates.\n *\n public object Delegates {\n /**\n * Returns a property delegate for a read/write
property with a non-`null` value that is initialized not during\n * object construction time but at a later time.
Trying to read the property before the initial value has been\n * assigned results in an exception.\n * \n *
@sample samples.properties.Delegates.notNullDelegate\n * \n public fun <T : Any> notNull():
ReadWriteProperty<Any?, T> = NotNullVar()\n\n /**\n * Returns a property delegate for a read/write property
that calls a specified callback function when changed.\n * @param initialValue the initial value of the property.\n
* @param onChange the callback which is called after the change of the property is made. The value of the
property\n * has already been changed when this callback is invoked.\n * \n * @sample
samples.properties.Delegates.observableDelegate\n * \n public inline fun <T> observable(initialValue: T,
crossinline onChange: (property: KProperty<*>, oldValue: T, newValue: T) -> Unit):\n
ReadWriteProperty<Any?, T> =\n object : ObservableProperty<T>(initialValue) {\n override fun
afterChange(property: KProperty<*>, oldValue: T, newValue: T) = onChange(property, oldValue, newValue)\n
}\n\n /**\n * Returns a property delegate for a read/write property that calls a specified callback function when
changed.\n * allowing the callback to veto the modification.\n * @param initialValue the initial value of the
property.\n * @param onChange the callback which is called before a change to the property value is attempted.\n
* The value of the property hasn't been changed yet, when this callback is invoked.\n * If the callback returns
`true` the value of the property is being set to the new value,\n * and if the callback returns `false` the new value
is discarded and the property remains its old value.\n * \n * @sample
samples.properties.Delegates.vetoableDelegate\n * \n @sample
samples.properties.Delegates.throwVetoableDelegate\n * \n public inline fun <T> vetoable(initialValue: T,
crossinline onChange: (property: KProperty<*>, oldValue: T, newValue: T) -> Boolean):\n
ReadWriteProperty<Any?, T> =\n object : ObservableProperty<T>(initialValue) {\n override fun
beforeChange(property: KProperty<*>, oldValue: T, newValue: T): Boolean = onChange(property, oldValue,
newValue)\n }\n\n private class NotNullVar<T : Any>(): ReadWriteProperty<Any?, T> {\n private var
value: T? = null\n\n public override fun getValue(thisRef: Any?, property: KProperty<*>): T {\n return value
?: throw IllegalStateException("Property ${property.name} should be initialized before get.")\n }\n\n public
override fun setValue(thisRef: Any?, property: KProperty<*>, value: T) {\n this.value = value\n
}\n}\n\n", "\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of
this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*\n\n package kotlin.properties\n\n import kotlin.reflect.KProperty\n\n /**\n * Base interface that can be used for
implementing property delegates of read-only properties.\n *\n * This is provided only for convenience; you don't

```

```

have to extend this interface
 * as long as your property delegate has methods with the same signatures.
 *
 * @param T the type of object which owns the delegated property.
 * @param V the type of the property value.
 *
 * \npublic fun interface ReadOnlyProperty<in T, out V> {
 *     /**
 *      * Returns the value of the property for the
 *      * given object.
 *      * @param thisRef the object for which the value is requested.
 *      * @param property the
 *      * metadata for the property.
 *      * @return the property value.
 *      */
 *     public operator fun getValue(thisRef: T,
 *     property: KProperty<*>): V
 * }
 *
 * \n\n/**
 * Base interface that can be used for implementing property delegates of
 * read-write properties.
 *
 * This is provided only for convenience; you don't have to extend this interface
 * as long as your property delegate has methods with the same signatures.
 *
 * @param T the type of object which
 * owns the delegated property.
 * @param V the type of the property value.
 *
 * \npublic interface
 * ReadWriteProperty<in T, V> : ReadOnlyProperty<T, V> {
 *     /**
 *      * Returns the value of the property for the
 *      * given object.
 *      * @param thisRef the object for which the value is requested.
 *      * @param property the
 *      * metadata for the property.
 *      * @return the property value.
 *      */
 *     public override operator fun
 *     getValue(thisRef: T, property: KProperty<*>): V
 *
 *     /**
 *      * Sets the value of the property for the given
 *      * object.
 *      * @param thisRef the object for which the value is requested.
 *      * @param property the metadata for
 *      * the property.
 *      * @param value the value to set.
 *      */
 *     public operator fun setValue(thisRef: T, property:
 *     KProperty<*>, value: V)
 * }
 *
 * \n\n/**
 * Base interface that can be used for implementing property delegate
 * providers.
 *
 * This is provided only for convenience; you don't have to extend this interface
 * as long as your
 * delegate provider has a method with the same signature.
 *
 * @param T the type of object which owns the
 * delegated property.
 * @param D the type of property delegates this provider provides.
 *
 * \n
 * @SinceKotlin("1.4")
 * \npublic fun interface PropertyDelegateProvider<in T, out D> {
 *     /**
 *      * Returns the
 *      * delegate of the property for the given object.
 *      *
 *      * This function can be used to extend the logic of creating
 *      * the object (e.g. perform validation checks)
 *      * to which the property implementation is delegated.
 *      *
 *      * @param thisRef the object for which property delegate is requested.
 *      * @param property the metadata for the
 *      * property.
 *      * @return the property delegate.
 *      */
 *     public operator fun provideDelegate(thisRef: T, property:
 *     KProperty<*>): D
 * }
 *
 * \n\n/**
 * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language
 * contributors.
 * Use of this source code is governed by the Apache 2.0 license that can be found in the
 * license/LICENSE.txt file.
 *
 * \n\npackage kotlin.properties
 * \n\nimport kotlin.reflect.KProperty
 *
 * \n\n/**
 * Implements the core logic of a property delegate for a read/write property that calls callback functions when
 * changed.
 *
 * @param initialValue the initial value of the property.
 *
 * \n\npublic abstract class
 * ObservableProperty<V>(initialValue: V) : ReadWriteProperty<Any?, V> {
 *     private var value = initialValue
 *
 *     /**
 *      * The callback which is called before a change to the property value is attempted.
 *      * The value of the
 *      * property hasn't been changed yet, when this callback is invoked.
 *      * If the callback returns `true` the value of the
 *      * property is being set to the new value,
 *      * and if the callback returns `false` the new value is discarded and the
 *      * property remains its old value.
 *      */
 *     protected open fun beforeChange(property: KProperty<*>, oldValue: V,
 *     newValue: V): Boolean = true
 *
 *     /**
 *      * The callback which is called after the change of the property is made.
 *      * The value of the property
 *      * has already been changed when this callback is invoked.
 *      */
 *     protected open
 *     fun afterChange(property: KProperty<*>, oldValue: V, newValue: V): Unit {}
 *
 *     public override fun
 *     getValue(thisRef: Any?, property: KProperty<*>): V {
 *         return value
 *     }
 *
 *     public override fun
 *     setValue(thisRef: Any?, property: KProperty<*>, value: V) {
 *         val oldValue = this.value
 *         if
 *         (!beforeChange(property, oldValue, value)) {
 *             return
 *         }
 *         this.value = value
 *         afterChange(property, oldValue, value)
 *     }
 * }
 *
 * \n\n/**
 * Copyright 2010-2020 JetBrains s.r.o. and Kotlin
 * Programming Language contributors.
 * Use of this source code is governed by the Apache 2.0 license that can be
 * found in the license/LICENSE.txt file.
 *
 * \n\n@file:Suppress("PackageDirectoryMismatch")
 * \n\npackage
 * kotlin
 * \n\nimport kotlin.reflect.*
 *
 * \n\n/**
 * An extension operator that allows delegating a read-only property of type
 * [V]
 * to a property reference to a property of type [V] or its subtype.
 *
 * @receiver A property reference to a
 * read-only or mutable property of type [V] or its subtype.
 *
 * The reference is without a receiver, i.e. it either
 * references a top-level property or
 * has the receiver bound to it.
 *
 * Example:
 *
 * ```
 * class Login(val
 * username: String)
 * val defaultLogin = Login("Admin")
 * val defaultUsername by defaultLogin::username
 *
 *

```

```

// equivalent to
val defaultUserName get() = defaultLogin.username
*/
@SinceKotlin("1.4")
@kotlin.internal.InlineOnly
public inline operator fun <V>
KProperty0<V>.getValue(thisRef: Any?, property: KProperty<*>): V {
    return get()
}
*/
An extension operator that allows delegating a mutable property of type [V] to a property reference to a mutable property of the same type [V].
*/
@receiver A property reference to a mutable property of type [V].
*/
The reference is without a receiver, i.e. it either references a top-level property or has the receiver bound to it.
*/
Example:
class Login(val username: String, var incorrectAttemptCounter: Int = 0)
val defaultLogin = Login("Admin")
var defaultLoginAttempts by defaultLogin::incorrectAttemptCounter
// equivalent to
var defaultLoginAttempts: Int
get() = defaultLogin.incorrectAttemptCounter
set(value) {
    defaultLogin.incorrectAttemptCounter = value
}
*/
@SinceKotlin("1.4")
@kotlin.internal.InlineOnly
public inline operator fun <V>
KMutableProperty0<V>.setValue(thisRef: Any?, property: KProperty<*>, value: V) {
    set(value)
}
*/
An extension operator that allows delegating a read-only member or extension property of type [V] to a property reference to a member or extension property of type [V] or its subtype.
*/
@receiver A property reference to a read-only or mutable property of type [V] or its subtype.
*/
The reference has an unbound receiver of type [T].
*/
Example:
class Login(val username: String)
val Login.user by Login::username
// equivalent to
val Login.user get() = this.username
*/
@SinceKotlin("1.4")
@kotlin.internal.InlineOnly
public inline operator fun <T, V> KProperty1<T, V>.getValue(thisRef: T, property: KProperty<*>): V {
    return get(thisRef)
}
*/
An extension operator that allows delegating a mutable member or extension property of type [V] to a property reference to a member or extension mutable property of the same type [V].
*/
@receiver A property reference to a read-only or mutable property of type [V] or its subtype.
*/
The reference has an unbound receiver of type [T].
*/
Example:
class Login(val username: String, var incorrectAttemptCounter: Int)
var Login.attempts by Login::incorrectAttemptCounter
// equivalent to
var Login.attempts: Int
get() = this.incorrectAttemptCounter
set(value) { this.incorrectAttemptCounter = value }
*/
@SinceKotlin("1.4")
@kotlin.internal.InlineOnly
public inline operator fun <T, V> KMutableProperty1<T, V>.setValue(thisRef: T, property: KProperty<*>, value: V) {
    set(thisRef, value)
}
*/
Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.
Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.
package kotlin.random
import kotlin.math.nextDown
*/
An abstract class that is implemented by random number generator algorithms.
*/
The companion object [Random.Default] is the default instance of [Random].
*/
To get a seeded instance of random generator use [Random] function.
*/
@sample samples.random.Randoms.defaultRandom
*/
@SinceKotlin("1.3")
public abstract class Random {
    /**
     * Gets the next random [bitCount] number of bits.
     * Generates an `Int` whose lower [bitCount] bits are filled with random values and the remaining upper bits are zero.
     * @param bitCount number of bits to generate, must be in range 0..32, otherwise the behavior is unspecified.
     * @sample samples.random.Randoms.nextBits
     */
    public abstract fun nextBits(bitCount: Int): Int
    /**
     * Gets the next random `Int` from the random number generator.
     * Generates an `Int` random value uniformly distributed between `Int.MIN_VALUE` and `Int.MAX_VALUE` (inclusive).
     * @sample samples.random.Randoms.nextInt
     */
    public open fun nextInt(): Int = nextBits(32)
    /**
     * Gets the next random non-negative `Int` from the random number generator less than the specified [until] bound.
     * Generates an `Int` random value uniformly distributed between `0` (inclusive) and the specified [until] bound (exclusive).
     * @param until must be positive.
     * @throws IllegalArgumentException if [until] is negative or zero.
     * @sample samples.random.Randoms.nextIntFromUntil
     */
    public open fun nextInt(until: Int): Int = nextInt(0, until)
    /**
     * Gets the next random `Int` from the random number generator in the specified range.
     * Generates an `Int` random value uniformly distributed between the specified [from] (inclusive) and [until] (exclusive) bounds.
     * @throws IllegalArgumentException if [from] is greater than or equal to [until].
     * @sample samples.random.Randoms.nextIntFromUntil
     */
    public open fun nextInt(from: Int, until:

```

```

Int): Int {
    checkRangeBounds(from, until)
    val n = until - from
    if (n > 0 || n == Int.MIN_VALUE)
    {
        val rnd = if (n and -n == n) {
            val bitCount = fastLog2(n)
            nextBits(bitCount)
        } else {
            var v: Int
            do {
                val bits = nextInt().ushr(1)
                v = bits % n
            } while (bits - v + (n - 1) < 0)
            v
        }
        return from + rnd
    } else {
        while (true) {
            val rnd = nextInt()
            if (rnd in from until until) return rnd
        }
    }
}

/**
 * Gets the next random `Long` from the random number generator.
 * Generates a `Long` random value uniformly distributed between `Long.MIN_VALUE` and `Long.MAX_VALUE` (inclusive).
 * @sample samples.random.Randoms.nextLong
 */
public open fun nextLong(): Long =
nextInt().toLong().shl(32) + nextInt()

/**
 * Gets the next random non-negative `Long` from the random number generator less than the specified [until] bound.
 * Generates a `Long` random value uniformly distributed between `0` (inclusive) and the specified [until] bound (exclusive).
 * @param until must be positive.
 * @throws IllegalArgumentException if [until] is negative or zero.
 * @sample samples.random.Randoms.nextLongFromUntil
 */
public open fun nextLong(until: Long): Long =
nextLong(0, until)

/**
 * Gets the next random `Long` from the random number generator in the specified range.
 * Generates a `Long` random value uniformly distributed between the specified [from] (inclusive) and [until] (exclusive) bounds.
 * @throws IllegalArgumentException if [from] is greater than or equal to [until].
 * @sample samples.random.Randoms.nextLongFromUntil
 */
public open fun nextLong(from: Long, until: Long): Long {
    checkRangeBounds(from, until)
    val n = until - from
    if (n > 0) {
        val rnd: Long
        if (n and -n == n) {
            val nLow = n.toInt()
            val nHigh = (n ushr 32).toInt()
            rnd = when {
                nLow != 0 -> {
                    val bitCount = fastLog2(nLow)
                    // toUInt().toLong()
                    nextBits(bitCount).toLong() and 0xFFFF_FFFF
                }
                nHigh == 1 -> {
                    // toUInt().toLong()
                    nextInt().toLong() and 0xFFFF_FFFF
                }
                else -> {
                    val bitCount = fastLog2(nHigh)
                    nextBits(bitCount).toLong().shl(32) + (nextInt().toLong() and 0xFFFF_FFFF)
                }
            }
        } else {
            var v: Long
            do {
                val bits = nextLong().ushr(1)
                v = bits % n
            } while (bits - v + (n - 1) < 0)
            rnd = v
        }
        return from + rnd
    } else {
        while (true) {
            val rnd = nextLong()
            if (rnd in from until until) return rnd
        }
    }
}

/**
 * Gets the next random [Boolean] value.
 * @sample samples.random.Randoms.nextBoolean
 */
public open fun nextBoolean(): Boolean = nextBits(1) != 0

/**
 * Gets the next random [Double] value uniformly distributed between 0 (inclusive) and 1 (exclusive).
 * @sample samples.random.Randoms.nextDouble
 */
public open fun nextDouble(): Double = doubleFromParts(nextBits(26), nextBits(27))

/**
 * Gets the next random non-negative `Double` from the random number generator less than the specified [until] bound.
 * Generates a `Double` random value uniformly distributed between 0 (inclusive) and [until] (exclusive).
 * @throws IllegalArgumentException if [until] is negative or zero.
 * @sample samples.random.Randoms.nextDoubleFromUntil
 */
public open fun nextDouble(until: Double): Double = nextDouble(0.0, until)

/**
 * Gets the next random `Double` from the random number generator in the specified range.
 * Generates a `Double` random value uniformly distributed between the specified [from] (inclusive) and [until] (exclusive) bounds.
 * [from] and [until] must be finite otherwise the behavior is unspecified.
 * @throws IllegalArgumentException if [from] is greater than or equal to [until].
 * @sample samples.random.Randoms.nextDoubleFromUntil
 */
public open fun nextDouble(from: Double, until: Double): Double {
    checkRangeBounds(from, until)
    val size = until - from
    val r = if (size.isInfinite() && from.isFinite() && until.isFinite()) {
        val r1 = nextDouble() * (until / 2 - from / 2)
        from + r1 + r1
    } else {
        from + nextDouble() * size
    }
    return if (r >= until) until.nextDown() else r
}

/**
 * Gets the next random [Float] value uniformly distributed between 0 (inclusive) and 1 (exclusive).
 * @sample samples.random.Randoms.nextFloat
 */
public open fun nextFloat(): Float = nextBits(24) / (1 shl 24).toFloat()

/**
 * Fills a subrange of the specified byte [array] starting from [fromIndex] inclusive and ending [toIndex] exclusive
 * with random bytes.
 * @return [array] with the subrange filled with

```

```

random bytes.\n * \n * @sample samples.random.Randoms.nextBytes\n * \n public open fun
nextBytes(array: ByteArray, fromIndex: Int = 0, toIndex: Int = array.size): ByteArray {\n
require(fromIndex in 0..array.size && toIndex in 0..array.size) { \"fromIndex ($fromIndex) or toIndex ($toIndex) are out of range:
0..${array.size}.\" }\n
require(fromIndex <= toIndex) { \"fromIndex ($fromIndex) must be not greater than
toIndex ($toIndex).\" }\n\n
val steps = (toIndex - fromIndex) / 4\n\n
var position = fromIndex\n
repeat(steps) {\n
val v = nextInt()\n
array[position] = v.toByte()\n
array[position + 1] =
v.ushr(8).toByte()\n
array[position + 2] = v.ushr(16).toByte()\n
array[position + 3] =
v.ushr(24).toByte()\n
position += 4\n
}\n\n
val remainder = toIndex - position\n
val vr =
nextInt(remainder * 8)\n
for (i in 0 until remainder) {\n
array[position + i] = vr.ushr(i * 8).toByte()\n
}\n\n
return array\n
}\n\n
/** \n * Fills the specified byte [array] with random bytes and returns it.\n * \n
* @return [array] filled with random bytes.\n * \n * @sample samples.random.Randoms.nextBytes\n * \n
public open fun nextBytes(array: ByteArray): ByteArray = nextBytes(array, 0, array.size)\n\n
/** \n * Creates a
byte array of the specified [size], filled with random bytes.\n * \n * @sample
samples.random.Randoms.nextBytes\n * \n public open fun nextBytes(size: Int): ByteArray =
nextBytes(ByteArray(size))\n\n
/** \n * The default random number generator.\n * \n * On JVM this
generator is thread-safe, its methods can be invoked from multiple threads.\n * \n * @sample
samples.random.Randoms.defaultRandom\n * \n companion object Default : Random(), Serializable {\n
private val defaultRandom: Random = defaultPlatformRandom()\n
private object Serialized : Serializable {\n
private const val serialVersionUID = 0L\n
private fun readResolve(): Any = Random\n
}\n\n
private fun writeReplace(): Any = Serialized\n\n
override fun nextBits(bitCount: Int): Int =
defaultRandom.nextBits(bitCount)\n
override fun nextInt(): Int = defaultRandom.nextInt()\n
override fun
nextInt(until: Int): Int = defaultRandom.nextInt(until)\n
override fun nextInt(from: Int, until: Int): Int =
defaultRandom.nextInt(from, until)\n\n
override fun nextLong(): Long = defaultRandom.nextLong()\n
override fun nextLong(until: Long): Long = defaultRandom.nextLong(until)\n
override fun nextLong(from:
Long, until: Long): Long = defaultRandom.nextLong(from, until)\n\n
override fun nextBoolean(): Boolean =
defaultRandom.nextBoolean()\n\n
override fun nextDouble(): Double = defaultRandom.nextDouble()\n
override fun
nextDouble(until: Double): Double = defaultRandom.nextDouble(until)\n
override fun
nextDouble(from: Double, until: Double): Double = defaultRandom.nextDouble(from, until)\n\n
override fun
nextFloat(): Float = defaultRandom.nextFloat()\n\n
override fun nextBytes(array: ByteArray): ByteArray =
defaultRandom.nextBytes(array)\n
override fun nextBytes(size: Int): ByteArray =
defaultRandom.nextBytes(size)\n
override fun nextBytes(array: ByteArray, fromIndex: Int, toIndex: Int):
ByteArray =\n
defaultRandom.nextBytes(array, fromIndex, toIndex)\n
}\n}\n\n
/** \n * Returns a repeatable
random number generator seeded with the given [seed] `Int` value.\n * \n * Two generators with the same seed
produce the same sequence of values within the same version of Kotlin runtime.\n * \n * *Note:* Future versions of
Kotlin may change the algorithm of this seeded number generator so that it will return\n * a sequence of values
different from the current one for a given seed.\n * \n * On JVM the returned generator is NOT thread-safe. Do not
invoke it from multiple threads without proper synchronization.\n * \n * @sample
samples.random.Randoms.seededRandom\n * \n @SinceKotlin(\"1.3\")\n\n
public fun Random(seed: Int): Random =
XorWowRandom(seed, seed.shr(31))\n\n
/** \n * Returns a repeatable random number generator seeded with the
given [seed] `Long` value.\n * \n * Two generators with the same seed produce the same sequence of values within
the same version of Kotlin runtime.\n * \n * *Note:* Future versions of Kotlin may change the algorithm of this
seeded number generator so that it will return\n * a sequence of values different from the current one for a given
seed.\n * \n * On JVM the returned generator is NOT thread-safe. Do not invoke it from multiple threads without
proper synchronization.\n * \n * @sample samples.random.Randoms.seededRandom\n
*\n
*\n
@SinceKotlin(\"1.3\")\n\n
public fun Random(seed: Long): Random = XorWowRandom(seed.toInt(),
seed.shr(32).toInt())\n\n
/** \n * Gets the next random `Int` from the random number generator in the specified
[range].\n * \n * Generates an `Int` random value uniformly distributed in the specified [range]:\n * from `range.start`
inclusive to `range.endInclusive` inclusive.\n * \n * @throws IllegalArgumentException if [range] is empty.\n

```



```

distributed between `0` (inclusive) and the specified [until] bound (exclusive).\n *\n * @throws
IllegalArgumentException if [until] is zero.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
Random.nextULong(until: ULong): ULong = nextULong(0uL, until)\n\n/**\n * Gets the next random [ULong] from
the random number generator in the specified range.\n *\n * Generates a [ULong] random value uniformly
distributed between the specified [from] (inclusive) and [until] (exclusive) bounds.\n *\n * @throws
IllegalArgumentException if [from] is greater than or equal to [until].\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
Random.nextULong(from: ULong, until: ULong): ULong {\n    checkULongRangeBounds(from, until)\n\n    val
signedFrom = from.toLong() xor Long.MIN_VALUE\n    val signedUntil = until.toLong() xor
Long.MIN_VALUE\n\n    val signedResult = nextLong(signedFrom, signedUntil) xor Long.MIN_VALUE\n
return signedResult.toULong()\n}\n\n/**\n * Gets the next random [ULong] from the random number generator in
the specified [range].\n *\n * Generates a [ULong] random value uniformly distributed in the specified [range]:\n *
from `range.start` inclusive to `range.endInclusive` inclusive.\n *\n * @throws IllegalArgumentException if [range]
is empty.\n *\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
Random.nextULong(range: ULongRange): ULong = when {\n    range.isEmpty() -> throw
IllegalArgumentException("Cannot get random in empty range: $range")\n    range.last < ULong.MAX_VALUE -
> nextULong(range.first, range.last + 1u)\n    range.first > ULong.MIN_VALUE -> nextULong(range.first - 1u,
range.last) + 1u\n    else -> nextULong()\n}\n\n/**\n * Fills the specified unsigned byte [array] with random bytes
and returns it.\n *\n * @return [array] filled with random bytes.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun Random.nextUBytes(array: UByteArray):
UByteArray {\n    nextBytes(array.asByteArray())\n    return array\n}\n\n/**\n * Creates an unsigned byte array of
the specified [size], filled with random bytes.\n *\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic
fun Random.nextUBytes(size: Int): UByteArray = nextBytes(size).asUByteArray()\n\n/**\n * Fills a subrange of the
specified `UByte` [array] starting from [fromIndex] inclusive and ending [toIndex] exclusive with random
UBytes.\n *\n * @return [array] with the subrange filled with random bytes.\n
*\n@SinceKotlin("1.3")\n@ExperimentalUnsignedTypes\npublic fun Random.nextUBytes(array: UByteArray,
fromIndex: Int = 0, toIndex: Int = array.size): UByteArray {\n    nextBytes(array.asByteArray(), fromIndex,
toIndex)\n    return array\n}\n\n\ninternal fun checkUIntRangeBounds(from: UInt, until: UInt) = require(until >
from) { boundsErrorMessage(from, until) }\ninternal fun checkULongRangeBounds(from: ULong, until: ULong) =
require(until > from) { boundsErrorMessage(from, until) }\n\n/**\n * Copyright 2010-2018 JetBrains s.r.o. and
Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that
can be found in the license/LICENSE.txt file.\n *\n@package kotlin.random\n\n/**\n * Random number generator,
using Marsaglia's "xorwow" algorithm\n *\n * Cycles after 2192 - 232 repetitions.\n *\n * For more details, see
Marsaglia, George (July 2003). "Xorshift RNGs". Journal of Statistical Software. 8 (14).
doi:10.18637/jss.v008.i14\n *\n * Available at https://www.jstatsoft.org/v08/i14/paper\n *\n *\ninternal class
XorWowRandom internal constructor(\n    private var x: Int,\n    private var y: Int,\n    private var z: Int,\n    private
var w: Int,\n    private var v: Int,\n    private var addend: Int\n) : Random(), Serializable {\n\n    internal
constructor(seed1: Int, seed2: Int) :n        this(seed1, seed2, 0, 0, seed1.inv(), (seed1 shl 10) xor (seed2 ushr
4))\n\n    init {\n        require((x or y or z or w or v) != 0) { "Initial state must have at least one non-zero element."
}\n\n        // some trivial seeds can produce several values with zeroes in upper bits, so we discard first 64\n
repeat(64) { nextInt() }\n    }\n\n    override fun nextInt(): Int {\n        // Equivalent to the xorwow algorithm\n        //
From Marsaglia, G. 2003. Xorshift RNGs. J. Statis. Soft. 8, 14, p. 5\n        var t = x\n        t = t xor (t ushr 2)\n        x
= y\n        y = z\n        z = w\n        val v0 = v\n        w = v0\n        t = (t xor (t shl 1)) xor v0 xor (v0 shl 4)\n        v =
t\n        addend += 362437\n        return t + addend\n    }\n\n    override fun nextBits(bitCount: Int): Int =\n
nextInt().takeUpperBits(bitCount)\n\n    private companion object {\n        private const val serialVersionUID: Long
= 0L\n    }\n}\n\n/**\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n *
Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n

```

```

*^/n/n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("\RangesKt")\n\npackage
kotlin.ranges\n/n/**\n * Represents a range of [Comparable] values.\n *^/nprivate open class ComparableRange<T :
Comparable<T>>(\n  override val start: T,\n  override val endInclusive: T\n) : ClosedRange<T> {\n\n  override
fun equals(other: Any?): Boolean {\n    return other is ComparableRange<*> && (isEmpty() && other.isEmpty())
||\n      start == other.start && endInclusive == other.endInclusive\n  }\n\n  override fun hashCode(): Int
{\n    return if (isEmpty()) -1 else 31 * start.hashCode() + endInclusive.hashCode()\n  }\n\n  override fun
toString(): String = "\$start..\$endInclusive"\n}\n/n/**\n * Creates a range from this [Comparable] value to the
specified [that] value.\n *^/n * This value needs to be smaller than or equal to [that] value, otherwise the returned
range will be empty.\n * @sample samples.ranges.Ranges.rangeFromComparable\n *^/npublic operator fun <T :
Comparable<T>> T.rangeTo(that: T): ClosedRange<T> = ComparableRange(this, that)\n/n/n/**\n * Represents a
range of floating point numbers.\n * Extends [ClosedRange] interface providing custom operation
[lessThanOrEquals] for comparing values of range domain type.\n *^/n * This interface is implemented by floating
point ranges returned by [Float.rangeTo] and [Double.rangeTo] operators to\n * achieve IEEE-754 comparison order
instead of total order of floating point numbers.\n *^/n@SinceKotlin("1.1")\npublic interface
ClosedFloatingPointRange<T : Comparable<T>> : ClosedRange<T> {\n  override fun contains(value: T): Boolean
= lessThanOrEquals(start, value) && lessThanOrEquals(value, endInclusive)\n  override fun isEmpty(): Boolean =
!lessThanOrEquals(start, endInclusive)\n\n  /**\n   * Compares two values of range domain type and returns true
if first is less than or equal to second.\n   *^/n   fun lessThanOrEquals(a: T, b: T): Boolean\n}\n/n/n/**\n * A
closed range of values of type `Double`.\n *^/n * Numbers are compared with the ends of this range according to
IEEE-754.\n *^/nprivate class ClosedDoubleRange(\n  start: Double,\n  endInclusive: Double\n) :
ClosedFloatingPointRange<Double> {\n  private val _start = start\n  private val _endInclusive = endInclusive\n
override val start: Double get() = _start\n  override val endInclusive: Double get() = _endInclusive\n\n  override
fun lessThanOrEquals(a: Double, b: Double): Boolean = a <= b\n  override fun contains(value: Double): Boolean
= value >= _start && value <= _endInclusive\n  override fun isEmpty(): Boolean = !(_start <= _endInclusive)\n\n
override fun equals(other: Any?): Boolean {\n    return other is ClosedDoubleRange && (isEmpty() &&
other.isEmpty()) ||\n      _start == other._start && _endInclusive == other._endInclusive\n  }\n\n  override
fun hashCode(): Int {\n    return if (isEmpty()) -1 else 31 * _start.hashCode() + _endInclusive.hashCode()\n  }\n\n
override fun toString(): String = "\$_start..$_endInclusive"\n}\n/n/n/**\n * Creates a range from this [Double]
value to the specified [that] value.\n *^/n * Numbers are compared with the ends of this range according to IEEE-
754.\n * @sample samples.ranges.Ranges.rangeFromDouble\n *^/n@SinceKotlin("1.1")\npublic operator fun
Double.rangeTo(that: Double): ClosedFloatingPointRange<Double> = ClosedDoubleRange(this, that)\n/n/n/**\n * A
closed range of values of type `Float`.\n *^/n * Numbers are compared with the ends of this range according to
IEEE-754.\n *^/nprivate class ClosedFloatRange(\n  start: Float,\n  endInclusive: Float\n) :
ClosedFloatingPointRange<Float> {\n  private val _start = start\n  private val _endInclusive = endInclusive\n
override val start: Float get() = _start\n  override val endInclusive: Float get() = _endInclusive\n\n  override
fun lessThanOrEquals(a: Float, b: Float): Boolean = a <= b\n  override fun contains(value: Float): Boolean = value
>= _start && value <= _endInclusive\n  override fun isEmpty(): Boolean = !(_start <= _endInclusive)\n\n
override fun equals(other: Any?): Boolean {\n    return other is ClosedFloatRange && (isEmpty() &&
other.isEmpty()) ||\n      _start == other._start && _endInclusive == other._endInclusive\n  }\n\n  override
fun hashCode(): Int {\n    return if (isEmpty()) -1 else 31 * _start.hashCode() + _endInclusive.hashCode()\n  }\n\n
override fun toString(): String = "\$_start..$_endInclusive"\n}\n/n/n/**\n * Creates a range from this [Float]
value to the specified [that] value.\n *^/n * Numbers are compared with the ends of this range according to IEEE-
754.\n * @sample samples.ranges.Ranges.rangeFromFloat\n *^/n@SinceKotlin("1.1")\npublic operator fun
Float.rangeTo(that: Float): ClosedFloatingPointRange<Float> = ClosedFloatRange(this, that)\n/n/n/**\n * Returns
`true` if this iterable range contains the specified [element].\n *^/n * Always returns `false` if the [element] is `null`.\n
*^/n@SinceKotlin("1.3")\n@kotlin.internal.InlineOnly\npublic inline operator fun <T, R> R.contains(element: T?):
Boolean where T : Any, R : Iterable<T>, R : ClosedRange<T> =\n  element != null &&
contains(element)\n/n/n\ninternal fun checkStepIsPositive(isPositive: Boolean, step: Number) {\n  if (!isPositive)

```

```

throw IllegalArgumentException("Step must be positive, was: $step.")\n\n"/*\n * Copyright 2010-2019
JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*/\n\n@file:kotlin.jvm.JvmName("KClasses")\n@file:Suppress("UNCHECKED_CAST")\n\npackage
kotlin.reflect\n\nimport kotlin.internal.LowPriorityInOverloadResolution\n\n/**\n * Casts the given [value] to the
class represented by this [KClass] object.\n * Throws an exception if the value is `null` or if it is not an instance of
this class.\n *\n * This is an experimental function that behaves as a similar function from kotlin.reflect.full on
JVM.\n *\n * @see [KClass.isInstance]\n * @see [KClass.safeCast]\n
*/\n\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@LowPriorityInOverloadResoluti
on\nfun <T : Any> KClass<T>.cast(value: Any?): T {\n    if (!isInstance(value)) throw ClassCastException("Value
cannot be cast to $qualifiedOrSimpleName")\n    return value as T\n}\n\n// TODO: replace with qualifiedName
when it is fully supported in K/JS\n\ninternal expect val KClass<*>.qualifiedOrSimpleName: String?\n\n/**\n * Casts
the given [value] to the class represented by this [KClass] object.\n * Returns `null` if the value is `null` or if it is not
an instance of this class.\n *\n * This is an experimental function that behaves as a similar function from
kotlin.reflect.full on JVM.\n *\n * @see [KClass.isInstance]\n * @see [KClass.cast]\n
*/\n\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@LowPriorityInOverloadResoluti
on\nfun <T : Any> KClass<T>.safeCast(value: Any?): T? {\n    return if (isInstance(value)) value as T else
null\n}\n\n"/*\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of
this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*/\n\npackage kotlin.reflect\n\nimport kotlin.jvm.JvmField\n\nimport kotlin.jvm.JvmStatic\n\n/**\n * Represents
a type projection. Type projection is usually the argument to another type in a type usage.\n * For example, in the
type `Array<out Number>`, `out Number` is the covariant projection of the type represented by the class
`Number`.\n *\n * Type projection is either the star projection, or an entity consisting of a specific type plus optional
variance.\n *\n * See the [Kotlin language documentation](https://kotlinlang.org/docs/reference/generics.html#type-
projections)\n * for more information.\n */\n\n@SinceKotlin("1.1")\n\npublic data class KTypeProjection
constructor(\n    /**\n     * The use-site variance specified in the projection, or `null` if this is a star projection.\n
*/\n    public val variance: KVariance?,\n    /**\n     * The type specified in the projection, or `null` if this is a star
projection.\n     */\n    public val type: KType?) {\n    init {\n        require((variance == null) == (type == null))\n
\n        if (variance == null)\n            "Star projection must have no type specified.\n" else\n
\n            "The projection variance $variance requires type to be specified.\n" }\n    }\n\n    override fun toString():
String = when (variance) {\n        null -> "*" \n        KVariance.INVARIANT -> type.toString()\n        KVariance.IN
-> "in $type"\n        KVariance.OUT -> "out $type"\n    }\n\n    public companion object {\n        //
provided for compiler access\n        @JvmField\n        @PublishedApi\n        internal val star: KTypeProjection =
KTypeProjection(null, null)\n\n        /**\n         * Star projection, denoted by the `*` character.\n         * For example,
in the type `KClass<*>`, `*` is the star projection.\n         * See the [Kotlin language
documentation](https://kotlinlang.org/docs/reference/generics.html#star-projections)\n         * for more
information.\n         */\n        public val STAR: KTypeProjection get() = star\n\n        /**\n         * Creates an
invariant projection of a given type. Invariant projection is just the type itself,\n         * without any use-site variance
modifiers applied to it.\n         * For example, in the type `Set<String>`, `String` is an invariant projection of the type
represented by the class `String`.\n         */\n        @JvmStatic\n        public fun invariant(type: KType):
KTypeProjection =\n            KTypeProjection(KVariance.INVARIANT, type)\n\n        /**\n         * Creates a
contravariant projection of a given type, denoted by the `in` modifier applied to a type.\n         * For example, in the
type `MutableList<in Number>`, `in Number` is a contravariant projection of the type of class `Number`.\n         */\n
        @JvmStatic\n        public fun contravariant(type: KType): KTypeProjection =\n            KTypeProjection(KVariance.IN, type)\n\n        /**\n         * Creates a covariant projection of a given type, denoted
by the `out` modifier applied to a type.\n         * For example, in the type `Array<out Number>`, `out Number` is a
covariant projection of the type of class `Number`.\n         */\n        @JvmStatic\n        public fun covariant(type:
KType): KTypeProjection =\n            KTypeProjection(KVariance.OUT, type)\n    }\n}\n\n"/*\n * Copyright 2010-

```

2019 JetBrains s.r.o. and Kotlin Programming Language contributors. Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.

`package kotlin.reflect`

Represents variance applied to a type parameter on the declaration site (\*declaration-site variance\*), or to a type in a projection (\*use-site variance\*).

See the [Kotlin language documentation](https://kotlinlang.org/docs/reference/generics.html#variance) for more information.

See [KTypeParameter.variance] and [KTypeProjection].

Since Kotlin 1.1

enum class KVariance {

    \* The affected type parameter or type is \*invariant\*, which means it has no variance applied to it.

    INVARIENT,

    \* The affected type parameter or type is \*contravariant\*. Denoted by the `in` modifier in the source code.

    IN,

    \* The affected type parameter or type is \*covariant\*. Denoted by the `out` modifier in the source code.

    OUT,

},

Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming Language contributors. Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.

Returns a runtime representation of the given reified type [T] as an instance of [KType]. Note that on JVM, the created type has no annotations ([KType.annotations] returns an empty list) even if the type in the source code is annotated. Support for type annotations might be added in a future version.

Since Kotlin 1.6

WasExperimental(ExperimentalStdlibApi::class)

public inline fun <reified T> typeOf(): KType = throw UnsupportedOperationException("This function is implemented as an intrinsic on all supported platforms.")

Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming Language contributors. Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.

file:kotlin.jvm.JvmMultifileClass

file:kotlin.jvm.JvmName("StringsKt")

package kotlin.text

An object to which char sequences and values can be appended.

next interface Appendable {

    \* Appends the specified character [value] to this Appendable and returns this instance.

    \* @param value the character to append.

    fun append(value: Char): Appendable

    \* Appends the specified character sequence [value] to this Appendable and returns this instance.

    \* @param value the character sequence to append. If [value] is `null`, then the four characters `"\null"` are appended to this Appendable.

    fun append(value: CharSequence?): Appendable

    \* Appends a subsequence of the specified character sequence [value] to this Appendable and returns this instance.

    \* @param value the character sequence from which a subsequence is appended. If [value] is `null`, then characters are appended as if [value] contained the four characters `"\null"`.

    \* @param startIndex the beginning (inclusive) of the subsequence to append.

    \* @param endIndex the end (exclusive) of the subsequence to append.

    \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of the [value] character sequence indices or when `startIndex > endIndex`.

    fun append(value: CharSequence?, startIndex: Int, endIndex: Int): Appendable

    \* Appends a subsequence of the specified character sequence [value] to this Appendable and returns this instance.

    \* @param value the character sequence from which a subsequence is appended.

    \* @param startIndex the beginning (inclusive) of the subsequence to append.

    \* @param endIndex the end (exclusive) of the subsequence to append.

    \* @throws IndexOutOfBoundsException or [IllegalArgumentException] when [startIndex] or [endIndex] is out of range of the [value] character sequence indices or when `startIndex > endIndex`.

    fun appendRange(value: CharSequence, startIndex: Int, endIndex: Int): T {

        @Suppress("UNCHECKED\_CAST")

        return append(value, startIndex, endIndex) as T

    }

    \* Appends all arguments to the given [Appendable].

    public fun <T : Appendable> T.append(vararg value: CharSequence?): T {

        for (item in value)

            append(item)

        return this

    }

    \* Appends a line feed character (`\n`) to this Appendable.

    Since Kotlin 1.4

    @kotlin.internal.InlineOnly

    public inline fun Appendable.appendLine(): Appendable = append("\n")

    \* Appends value to the given Appendable and a line feed character (`\n`) after it.

    Since Kotlin 1.4

    @kotlin.internal.InlineOnly

    public inline fun Appendable.appendLine(value: CharSequence?): Appendable = append(value).appendLine()

    \* Appends value

to the given Appendable and a line feed character (`\n`) after it.

```
*\n@SinceKotlin("1.4")\n@kotlin.internal.InlineOnly\npublic inline fun Appendable.appendLine(value: Char):  
Appendable = append(value).appendLine()\n\ninternal fun <T> Appendable.appendElement(element: T,  
transform: ((T) -> CharSequence)?) {\n    when {\n        transform != null -> append(transform(element))\n        element is CharSequence? -> append(element)\n        element is Char -> append(element)\n        else ->  
        append(element.toString())\n    }\n}\n\n"/*\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming  
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the  
license/LICENSE.txt file.\n\n*\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("StringsKt")\n\npackage  
kotlin.text\n\n/**\n * Trims leading whitespace characters followed by [marginPrefix] from every line of a source  
string and removes\n * the first and the last lines if they are blank (notice difference blank vs empty).\n * Doesn't\n * affect a line if it doesn't contain [marginPrefix] except the first and the last blank lines.\n * Doesn't preserve the  
original line endings.\n * @param marginPrefix non-blank string, which is used as a margin delimiter. Default is  
`|` (pipe character).\n * @sample samples.text.Strings.trimMargin\n * @see trimIndent\n * @see  
kotlin.text.isWhitespace\n *\npublic fun String.trimMargin(marginPrefix: String = "|"): String =\n    replaceIndentByMargin("|", marginPrefix)\n\n/**\n * Detects indent by [marginPrefix] as it does [trimMargin] and  
replace it with [newIndent].\n * @param marginPrefix non-blank string, which is used as a margin delimiter.  
Default is `|` (pipe character).\n *\npublic fun String.replaceIndentByMargin(newIndent: String = "|",  
marginPrefix: String = "|"): String {\n    require(marginPrefix.isNotBlank()) { "marginPrefix must be non-blank  
string." }\n    val lines = lines()\n    return lines.reindent(length + newIndent.length * lines.size,  
getIndentFunction(newIndent), { line ->\n        val firstNonWhitespaceIndex = line.indexOfFirst { !it.isWhitespace() }\n    }\n    when {\n        firstNonWhitespaceIndex == -1 -> null\n        line.startsWith(marginPrefix,  
firstNonWhitespaceIndex) -> line.substring(firstNonWhitespaceIndex + marginPrefix.length)\n        else -> null\n    }\n})\n\n/**\n * Detects a common minimal indent of all the input lines, removes it from every line and  
also removes the first and the last\n * lines if they are blank (notice difference blank vs empty).\n * Note that  
blank lines do not affect the detected indent level.\n * In case if there are non-blank lines with no leading  
whitespace characters (no indent at all) then the\n * common indent is 0, and therefore this function doesn't change  
the indentation.\n * Doesn't preserve the original line endings.\n * @sample  
samples.text.Strings.trimIndent\n * @see trimMargin\n * @see kotlin.text.isBlank\n *\npublic fun  
String.trimIndent(): String = replaceIndent("|")\n\n/**\n * Detects a common minimal indent like it does  
[trimIndent] and replaces it with the specified [newIndent].\n *\npublic fun String.replaceIndent(newIndent: String  
= "|"): String {\n    val lines = lines()\n    val minCommonIndent = lines\n        .filter(String::isNotBlank)\n        .map(String::indentWidth)\n        .minOrNull() ?: 0\n    return lines.reindent(length + newIndent.length *  
lines.size, getIndentFunction(newIndent), { line -> line.drop(minCommonIndent) })\n}\n\n/**\n * Prepends [indent]  
to every line of the original string.\n * Doesn't preserve the original line endings.\n *\npublic fun  
String.prependIndent(indent: String = "| "): String =\n    lineSequence()\n        .map {\n            when {\n                it.isBlank() -> {\n                    when {\n                        it.length < indent.length -> indent\n                        else -> it\n                    }\n                }\n                else -> indent + it\n            }\n        }\n        .joinToString("\n")\n\nprivate fun  
String.indentWidth(): Int = indexOfFirst { !it.isWhitespace() }.let { if (it == -1) length else it }\n\nprivate fun  
getIndentFunction(indent: String) = when {\n    indent.isEmpty() -> { line: String -> line }\n    else -> { line: String -  
> indent + line }\n}\n\nprivate inline fun List<String>.reindent(\n    resultSizeEstimate: Int,\n    indentAddFunction:  
(String) -> String,\n    indentCutFunction: (String) -> String?): String {\n    val lastIndex = lastIndex\n    return  
mapIndexedNotNull { index, value ->\n        if ((index == 0 || index == lastIndex) && value.isBlank())\n            null\n        else\n            indentCutFunction(value)?.let(indentAddFunction)?: value\n    }\n    .joinTo(StringBuilder(resultSizeEstimate), "\n")\n    .toString()\n}\n\n"/*\n * Copyright 2010-2018 JetBrains  
s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0  
license that can be found in the license/LICENSE.txt file.\n *\npackage kotlin.text\n\n/**\n * Defines names for  
Unicode symbols used in proper Typography.\n *\npublic object Typography {\n    /** The character &#x22;
```

```

\u2013 quotation mark */\n public const val quote: Char = "\u0022"\n /** The character &#x24; \u2013 dollar
sign */\n public const val dollar: Char = "\u0024"\n /** The character &#x26; \u2013 ampersand */\n public
const val amp: Char = "\u0026"\n /** The character &#x3C; \u2013 less-than sign */\n public const val less:
Char = "\u003C"\n /** The character &#x3E; \u2013 greater-than sign */\n public const val greater: Char =
"\u003E"\n /** The non-breaking space character */\n public const val nbsp: Char = "\u00A0"\n /** The
character &#xD7; */\n public const val times: Char = "\u00D7"\n /** The character &#xA2; */\n public const
val cent: Char = "\u00A2"\n /** The character &#xA3; */\n public const val pound: Char = "\u00A3"\n /** The
character &#xA7; */\n public const val section: Char = "\u00A7"\n /** The character &#xA9; */\n public const
val copyright: Char = "\u00A9"\n /** The character &#xAB; */\n @SinceKotlin("1.6")\n public const val
leftGuillemet: Char = "\u00AB"\n /** The character &#xBB; */\n @SinceKotlin("1.6")\n public const val
rightGuillemet: Char = "\u00BB"\n /** The character &#xAE; */\n public const val registered: Char =
"\u00AE"\n /** The character &#xB0; */\n public const val degree: Char = "\u00B0"\n /** The character
&#xB1; */\n public const val plusMinus: Char = "\u00B1"\n /** The character &#xB6; */\n public const val
paragraph: Char = "\u00B6"\n /** The character &#xB7; */\n public const val middleDot: Char = "\u00B7"\n
/** The character &#xBD; */\n public const val half: Char = "\u00BD"\n /** The character &#x2013; */\n
public const val ndash: Char = "\u2013"\n /** The character &#x2014; */\n public const val mdash: Char =
"\u2014"\n /** The character &#x2018; */\n public const val leftSingleQuote: Char = "\u2018"\n /** The
character &#x2019; */\n public const val rightSingleQuote: Char = "\u2019"\n /** The character &#x201A; */\n
public const val lowSingleQuote: Char = "\u201A"\n /** The character &#x201C; */\n public const val
leftDoubleQuote: Char = "\u201C"\n /** The character &#x201D; */\n public const val rightDoubleQuote: Char
= "\u201D"\n /** The character &#x201E; */\n public const val lowDoubleQuote: Char = "\u201E"\n /** The
character &#x2020; */\n public const val dagger: Char = "\u2020"\n /** The character &#x2021; */\n public
const val doubleDagger: Char = "\u2021"\n /** The character &#x2022; */\n public const val bullet: Char =
"\u2022"\n /** The character &#x2026; */\n public const val ellipsis: Char = "\u2026"\n /** The character
&#x2032; */\n public const val prime: Char = "\u2032"\n /** The character &#x2033; */\n public const val
doublePrime: Char = "\u2033"\n /** The character &#x20AC; */\n public const val euro: Char = "\u20AC"\n
/** The character &#x2122; */\n public const val tm: Char = "\u2122"\n /** The character &#x2248; */\n
public const val almostEqual: Char = "\u2248"\n /** The character &#x2260; */\n public const val notEqual:
Char = "\u2260"\n /** The character &#x2264; */\n public const val lessOrEqual: Char = "\u2264"\n /** The
character &#x2265; */\n public const val greaterOrEqual: Char = "\u2265"\n /** The character &#xAB; */\n
@Deprecated("This constant has a typo in the name. Use leftGuillemet instead."),
ReplaceWith("Typography.leftGuillemet("))\n @DeprecatedSinceKotlin("1.6")\n public const val
leftGuillemete: Char = "\u00AB"\n /** The character &#xBB; */\n @Deprecated("This constant has a typo in
the name. Use rightGuillemet instead."), ReplaceWith("Typography.rightGuillemet("))\n
@DeprecatedSinceKotlin("1.6")\n public const val rightGuillemete: Char = "\u00BB"\n }"/*\n * Copyright
2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed
by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage kotlin.text\n\n/**\n *
Represents a collection of captured groups in a single match of a regular expression.\n *\n * This collection has size
of `groupCount + 1` where `groupCount` is the count of groups in the regular expression.\n *\n * Groups are indexed
from 1 to `groupCount` and group with the index 0 corresponds to the entire match.\n *\n * An element of the
collection at the particular index can be `null`,\n *\n * if the corresponding group in the regular expression is optional
and\n *\n * there was no match captured by that group.\n */\n\npublic interface MatchGroupCollection :
Collection<MatchGroup?> {\n\n /** Returns a group with the specified [index].\n *\n * @return An instance
of [MatchGroup] if the group with the specified [index] was matched or `null` otherwise.\n *\n * Groups are
indexed from 1 to the count of groups in the regular expression. A group with the index 0\n *\n * corresponds to the
entire match.\n */\n\n public operator fun get(index: Int): MatchGroup?\n\n /**\n * Extends
[MatchGroupCollection] by introducing a way to get matched groups by name, when regex supports it.\n
*/\n\n @SinceKotlin("1.1")\n\n public interface MatchNamedGroupCollection : MatchGroupCollection {\n\n /**\n *

```

```

Returns a named group with the specified [name].\n * @return An instance of [MatchGroup] if the group with the
specified [name] was matched or `null` otherwise.\n * @throws IllegalArgumentException if there is no group
with the specified [name] defined in the regex pattern.\n * @throws UnsupportedOperationException if getting
named groups isn't supported on the current platform.\n * ^\n public operator fun get(name: String):
MatchGroup?\n}\n\n/**\n * Represents the results from a single regular expression match.\n * ^\npublic interface
MatchResult {\n /** The range of indices in the original string where match was captured. *\n public val range:
IntRange\n /** The substring from the input string captured by this match. *\n public val value: String\n /**\n * A collection of groups matched by the regular expression.\n * ^\n * This collection has size of `groupCount +
1` where `groupCount` is the count of groups in the regular expression.\n * Groups are indexed from 1 to
`groupCount` and group with the index 0 corresponds to the entire match.\n * ^\n public val groups:
MatchGroupCollection\n /**\n * A list of matched indexed group values.\n * ^\n * This list has size of
`groupCount + 1` where `groupCount` is the count of groups in the regular expression.\n * Groups are indexed
from 1 to `groupCount` and group with the index 0 corresponds to the entire match.\n * ^\n * If the group in the
regular expression is optional and there were no match captured by that group,\n * corresponding item in
[groupValues] is an empty string.\n * ^\n * @sample
samples.text.Regexps.matchDestructuringToGroupValues\n * ^\n public val groupValues: List<String>\n\n
/**\n * An instance of [MatchResult.Destructured] wrapper providing components for destructuring assignment
of group values.\n * ^\n * component1 corresponds to the value of the first group, component2 \u2014 of the
second, and so on.\n * ^\n * @sample samples.text.Regexps.matchDestructuringToGroupValues\n * ^\n
public val destructured: Destructured get() = Destructured(this)\n\n /** Returns a new [MatchResult] with the
results for the next match, starting at the position\n * at which the last match ended (at the character after the last
matched character).\n * ^\n public fun next(): MatchResult?\n}\n\n /**\n * Provides components for
destructuring assignment of group values.\n * ^\n * [component1] corresponds to the value of the first group,
[component2] \u2014 of the second, and so on.\n * ^\n * If the group in the regular expression is optional and
there were no match captured by that group,\n * corresponding component value is an empty string.\n * ^\n *
@sample samples.text.Regexps.matchDestructuringToGroupValues\n * ^\n public class Destructured internal
constructor(public val match: MatchResult) {\n @kotlin.internal.InlineOnly\n public operator inline fun
component1(): String = match.groupValues[1]\n @kotlin.internal.InlineOnly\n public operator inline fun
component2(): String = match.groupValues[2]\n @kotlin.internal.InlineOnly\n public operator inline fun
component3(): String = match.groupValues[3]\n @kotlin.internal.InlineOnly\n public operator inline fun
component4(): String = match.groupValues[4]\n @kotlin.internal.InlineOnly\n public operator inline fun
component5(): String = match.groupValues[5]\n @kotlin.internal.InlineOnly\n public operator inline fun
component6(): String = match.groupValues[6]\n @kotlin.internal.InlineOnly\n public operator inline fun
component7(): String = match.groupValues[7]\n @kotlin.internal.InlineOnly\n public operator inline fun
component8(): String = match.groupValues[8]\n @kotlin.internal.InlineOnly\n public operator inline fun
component9(): String = match.groupValues[9]\n @kotlin.internal.InlineOnly\n public operator inline fun
component10(): String = match.groupValues[10]\n\n /**\n * Returns destructured group values as a list of
strings.\n * ^\n * First value in the returned list corresponds to the value of the first group, and so on.\n * ^\n
* @sample samples.text.Regexps.matchDestructuringToGroupValues\n * ^\n public fun toList():
List<String> = match.groupValues.subList(1, match.groupValues.size)\n }\n}\n\n /**\n * Copyright 2010-2021
JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n\n
* ^\n\n@file:kotlin.jvm.JvmMultifileClass()\n@file:kotlin.jvm.JvmName("DurationUnitKt")\n\npackage
kotlin.time\n\n/**\n * The list of possible time measurement units, in which a duration can be expressed.\n * ^\n *
The smallest time unit is [NANOSECONDS] and the largest is [DAYS], which corresponds to exactly 24
[HOURS].\n * ^\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalTime::class)\npublic expect enum class
DurationUnit {\n /**\n * Time unit representing one nanosecond, which is 1/1000 of a microsecond.\n * ^\n
NANOSECONDS,\n /**\n * Time unit representing one microsecond, which is 1/1000 of a millisecond.\n * ^\n

```

```

*/\n MICROSECONDS,\n /**\n * Time unit representing one millisecond, which is 1/1000 of a second.\n
*/\n MILLISECONDS,\n /**\n * Time unit representing one second.\n */\n SECONDS,\n /**\n *
Time unit representing one minute.\n */\n MINUTES,\n /**\n * Time unit representing one hour.\n */\n
HOURS,\n /**\n * Time unit representing one day, which is always equal to 24 hours.\n */\n
DAYS;\n}\n\n/** Converts the given time duration [value] expressed in the specified [sourceUnit] into the specified
[targetUnit]. */\n@SinceKotlin("1.3")\ninternal expect fun convertDurationUnit(value: Double, sourceUnit:
DurationUnit, targetUnit: DurationUnit): Double\n// overflown result is
unspecified\n@SinceKotlin("1.5")\ninternal expect fun convertDurationUnitOverflow(value: Long, sourceUnit:
DurationUnit, targetUnit: DurationUnit): Long\n// overflown result is coerced in the Long range
boundaries\n@SinceKotlin("1.5")\ninternal expect fun convertDurationUnit(value: Long, sourceUnit:
DurationUnit, targetUnit: DurationUnit):
Long\n\n\n@SinceKotlin("1.3")\n@Suppress("REDUNDANT_ELSE_IN_WHEN")\ninternal fun
DurationUnit.shortName(): String = when (this) {\n DurationUnit.NANOSECONDS -> "ns"\n
DurationUnit.MICROSECONDS -> "us"\n DurationUnit.MILLISECONDS -> "ms"\n
DurationUnit.SECONDS -> "s"\n DurationUnit.MINUTES -> "m"\n DurationUnit.HOURS -> "h"\n
DurationUnit.DAYS -> "d"\n else -> error("Unknown unit: $this")\n}\n\n@SinceKotlin("1.5")\ninternal fun
durationUnitByShortName(shortName: String): DurationUnit = when (shortName) {\n "ns" ->
DurationUnit.NANOSECONDS\n "us" -> DurationUnit.MICROSECONDS\n "ms" ->
DurationUnit.MILLISECONDS\n "s" -> DurationUnit.SECONDS\n "m" -> DurationUnit.MINUTES\n
"h" -> DurationUnit.HOURS\n "d" -> DurationUnit.DAYS\n else -> throw
IllegalArgumentExcepion("Unknown duration unit short name:
$shortName")\n}\n\n@SinceKotlin("1.5")\ninternal fun durationUnitByIsoChar(isoChar: Char,
isTimeComponent: Boolean): DurationUnit =\n when {\n !isTimeComponent -> {\n when (isoChar)
{\n 'D' -> DurationUnit.DAYS\n else -> throw IllegalArgumentExcepion("Invalid or
unsupported duration ISO non-time unit: $isoChar")\n }\n } else -> {\n when (isoChar) {\n
'H' -> DurationUnit.HOURS\n 'M' -> DurationUnit.MINUTES\n 'S' ->
DurationUnit.SECONDS\n else -> throw IllegalArgumentExcepion("Invalid duration ISO time unit:
$isoChar")\n }\n }\n },""/\n * Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\npackage kotlin.time\n\nimport kotlin.annotation.AnnotationTarget.*\n
*\n * This annotation marks the experimental preview of the standard library API for measuring time and working with
durations.\n * > Note that this API is in a preview state and has a very high chance of being changed in the
future.\n * Do not use it if you develop a library since your library will become binary incompatible\n * with the
future versions of the standard library.\n * Any usage of a declaration annotated with `@ExperimentalTime`
must be accepted either by\n * annotating that usage with the [OptIn] annotation, e.g.
`@OptIn(ExperimentalTime::class)`,\n * or by using the compiler argument `-Xopt-in=kotlin.time.ExperimentalTime`.
\n */\n@Suppress("DEPRECATION")\n@Experimental(level =
Experimental.Level.ERROR)\n@RequiresOptIn(level =
RequiresOptIn.Level.ERROR)\n@MustBeDocumented\n@Retention(AnnotationRetention.BINARY)\n@Target(\n
CLASS,\n ANNOTATION_CLASS,\n PROPERTY,\n FIELD,\n LOCAL_VARIABLE,\n
VALUE_PARAMETER,\n CONSTRUCTOR,\n FUNCTION,\n PROPERTY_GETTER,\n
PROPERTY_SETTER,\n TYPEALIAS)\n@SinceKotlin("1.3")\npublic annotation class
ExperimentalTime\n"/\n * Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\npackage kotlin.time\n\n/**\n * A source of time for measuring time intervals.\n
*\n * The only operation provided by the time source is [markNow]. It returns a [TimeMark], which can be used to
query the elapsed time later.\n * @see [measureTime]\n * @see [measureTimedValue]\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalTime\npublic interface TimeSource {\n /**\n * Marks a point in

```

```

time on this time source.\n    *\n    * The returned [TimeMark] instance encapsulates the captured time point and
allows querying\n    * the duration of time interval [elapsed][TimeMark.elapsedNow] from that point.\n    */\n
public fun markNow(): TimeMark\n\n    /**\n    * The most precise time source available in the platform.\n    *\n    * This time source returns its readings from a source of monotonic time when it is available in a target platform,\n    * and resorts to a non-monotonic time source otherwise.\n    */\n    public object Monotonic : TimeSource by
MonotonicTimeSource {\n        override fun toString(): String = MonotonicTimeSource.toString()\n    }\n\n    public companion object {\n\n        /**\n        * Represents a time point notched on a particular [TimeSource].
Remains bound to the time source it was taken from\n        * and allows querying for the duration of time elapsed from
that point (see the function [elapsedNow]).\n        */\n        @SinceKotlin("1.3")\n        @ExperimentalTime\n        public abstract class
TimeMark {\n            /**\n            * Returns the amount of time passed from this mark measured with the time source from
which this mark was taken.\n            *\n            * Note that the value returned by this function can change on subsequent
invocations.\n            */\n            public abstract fun elapsedNow(): Duration\n\n            /**\n            * Returns a time mark on the same
time source that is ahead of this time mark by the specified [duration].\n            *\n            * The returned time mark is more
_late_ when the [duration] is positive, and more _early_ when the [duration] is negative.\n            */\n            public open
operator fun plus(duration: Duration): TimeMark = AdjustedTimeMark(this, duration)\n\n            /**\n            * Returns a
time mark on the same time source that is behind this time mark by the specified [duration].\n            *\n            * The
returned time mark is more _early_ when the [duration] is positive, and more _late_ when the [duration] is
negative.\n            */\n            public open operator fun minus(duration: Duration): TimeMark = plus(-duration)\n\n            /**\n            * Returns true if this time mark has passed according to the time source from which this mark was taken.\n            *\n            * Note that the value returned by this function can change on subsequent invocations.\n            * If the time source is
monotonic, it can change only from `false` to `true`, namely, when the time mark becomes behind the current point
of the time source.\n            */\n            public fun hasPassedNow(): Boolean = !elapsedNow().isNegative()\n\n            /**\n            * Returns false if this time mark has not passed according to the time source from which this mark was taken.\n            *\n            * Note that the value returned by this function can change on subsequent invocations.\n            * If the time source is
monotonic, it can change only from `true` to `false`, namely, when the time mark becomes behind the current point
of the time source.\n            */\n            public fun hasNotPassedNow(): Boolean =
elapsedNow().isNegative()\n\n        }\n\n        @ExperimentalTime\n        @SinceKotlin("1.3")\n        @kotlin.internal.InlineOnly\n        @Deprecated(\n            "Subtracting one TimeMark from another is not a well defined operation because these time marks
could have been obtained from the different time sources.",\n            level =
DeprecationLevel.ERROR\n        )\n        @Suppress("UNUSED_PARAMETER")\n        public inline operator fun
TimeMark.minus(other: TimeMark): Duration = throw Error("Operation is
disallowed.")\n\n        @ExperimentalTime\n        @SinceKotlin("1.3")\n        @kotlin.internal.InlineOnly\n        @Deprecated(\n            "Comparing one TimeMark to another is not a well defined operation because these time marks could have been
obtained from the different time sources.",\n            level =
DeprecationLevel.ERROR\n        )\n        @Suppress("UNUSED_PARAMETER")\n        public inline operator fun
TimeMark.compareTo(other: TimeMark): Int = throw Error("Operation is
disallowed.")\n\n        @ExperimentalTime\n        private class AdjustedTimeMark(val mark: TimeMark, val adjustment:
Duration) : TimeMark() {\n            override fun elapsedNow(): Duration = mark.elapsedNow() - adjustment\n\n            override fun plus(duration: Duration): TimeMark = AdjustedTimeMark(mark, adjustment + duration)\n        }\n    }\n}
/**\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is
governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\npackage
kotlin.time\n\n@SinceKotlin("1.3")\n@ExperimentalTime\ninternal expect object MonotonicTimeSource :
TimeSource\n\n/**\n * An abstract class used to implement time sources that return their readings as [Long] values
in the specified [unit].\n */\n * @property unit The unit in which this time source's readings are expressed.\n
*/\n\n@SinceKotlin("1.3")\n@ExperimentalTime\npublic abstract class AbstractLongTimeSource(protected val
unit: DurationUnit) : TimeSource {\n    /**\n    * This protected method should be overridden to return the current
reading of the time source expressed as a [Long] number\n    * in the unit specified by the [unit] property.\n    */\n    protected abstract fun read(): Long\n\n    private class LongTimeMark(private val startedAt: Long, private val

```

```

timeSource: AbstractLongTimeSource, private val offset: Duration) : TimeMark() {
    override fun
    elapsedNow(): Duration = (timeSource.read() - startedAt).toDuration(timeSource.unit) - offset
    override fun
    plus(duration: Duration): TimeMark = LongTimeMark(startedAt, timeSource, offset + duration)
}

/** An abstract class used to
implement time sources that return their readings as [Double] values in the specified [unit].
@property unit
The unit in which this time source's readings are expressed.

@SinceKotlin("1.3")
@ExperimentalTime
public abstract class AbstractDoubleTimeSource(protected val
unit: DurationUnit) : TimeSource {
    /** This protected method should be overridden to return the current
reading of the time source expressed as a [Double] number
    * in the unit specified by the [unit] property.
    protected abstract fun read(): Double
    private class DoubleTimeMark(private val startedAt: Double,
private val timeSource: AbstractDoubleTimeSource, private val offset: Duration) : TimeMark() {
    override fun
    elapsedNow(): Duration = (timeSource.read() - startedAt).toDuration(timeSource.unit) - offset
    override fun
    plus(duration: Duration): TimeMark = DoubleTimeMark(startedAt, timeSource, offset + duration)
}

/** A time source
that has programmatically updatable readings. It is useful as a predictable source of time in tests.
@property unit
The current
reading value can be advanced by the specified duration amount with the operator [plusAssign]:
val
timeSource = TestTimeSource()
timeSource += 10.seconds
Implementation note: the current
reading value is stored as a [Long] number of nanoseconds,
thus it's capable to represent a time range of
approximately 1292 years.
Should the reading value overflow as the result of [plusAssign] operation, an
[IllegalStateException] is thrown.

@SinceKotlin("1.3")
@ExperimentalTime
public class TestTimeSource
: AbstractLongTimeSource(unit = DurationUnit.NANOSECONDS) {
    private var reading: Long = 0L
    override fun read(): Long = reading
    /** Advances the current reading value of this time source by the
specified [duration].
    * [duration] value is rounded down towards zero when converting it to a [Long]
number of nanoseconds.
    * For example, if the duration being added is `0.6.nanoseconds`, the reading doesn't
advance because
    * the duration value is rounded to zero nanoseconds.
    * @throws
IllegalStateException when the reading value overflows as the result of this operation.
    public operator fun
    plusAssign(duration: Duration) {
        val longDelta = duration.toLong(unit)
        reading = if (longDelta !=
        Long.MIN_VALUE && longDelta != Long.MAX_VALUE) {
            // when delta fits in long, add it as long
            val newReading = reading + longDelta
            if (reading xor longDelta >= 0 && reading xor newReading < 0)
                overflow(duration)
            newReading
        } else {
            val delta = duration.toDouble(unit)
            // when
            delta is greater than long, add it as double
            val newReading = reading + delta
            if (newReading >
            Long.MAX_VALUE || newReading < Long.MIN_VALUE) overflow(duration)
            newReading.toLong()
        }
    }
    private fun overflow(duration: Duration) {
        throw IllegalStateException("TestTimeSource will
        overflow if its reading ${reading}ns is advanced by $duration.")
    }
}

/** Copyright 2010-2020 JetBrains
s.r.o. and Kotlin Programming Language contributors.
Use of this source code is governed by the Apache 2.0
license that can be found in the license/LICENSE.txt file.

package kotlin.time
import
kotlin.contracts.*
/** Executes the given function [block] and returns the duration of elapsed time interval.
@property unit
The elapsed time is measured with [TimeSource.Monotonic].

@SinceKotlin("1.3")
@ExperimentalTime
public inline fun measureTime(block: () -> Unit): Duration {
    contract {
        callsInPlace(block, InvocationKind.EXACTLY_ONCE)
    }
    return
    TimeSource.Monotonic.measureTime(block)
}

/** Executes the given function [block] and returns the
duration of elapsed time interval.
@property unit
The elapsed time is measured with the specified `this` [TimeSource]
instance.

@SinceKotlin("1.3")
@ExperimentalTime
public inline fun TimeSource.measureTime(block: ()
-> Unit): Duration {
    contract {
        callsInPlace(block, InvocationKind.EXACTLY_ONCE)
    }
    val
    mark = markNow()
    block()
    return mark.elapsedNow()
}

/** Data class representing a result of
executing an action, along with the duration of elapsed time interval.
@property value
the result of the
action.
@property duration
the time elapsed to execute the action.

@SinceKotlin("1.3")
@ExperimentalTime
public data class TimedValue<T>(val value: T, val duration:

```

```

Duration)\n\n/**\n * Executes the given function [block] and returns an instance of [TimedValue] class, containing
both\n * the result of the function execution and the duration of elapsed time interval.\n *\n * The elapsed time is
measured with [TimeSource.Monotonic].\n *\n@SinceKotlin("1.3")\n@ExperimentalTime\npublic inline fun <T>
measureTimedValue(block: () -> T): TimedValue<T> {\n    contract {\n        callsInPlace(block,
InvocationKind.EXACTLY_ONCE)\n    }\n\n    return
TimeSource.Monotonic.measureTimedValue(block)\n}\n\n/**\n * Executes the given [block] and returns an
instance of [TimedValue] class, containing both\n * the result of function execution and the duration of elapsed time
interval.\n *\n * The elapsed time is measured with the specified `this` [TimeSource] instance.\n
*\n@SinceKotlin("1.3")\n@ExperimentalTime\npublic inline fun <T> TimeSource.measureTimedValue(block: ()
-> T): TimedValue<T> {\n    contract {\n        callsInPlace(block, InvocationKind.EXACTLY_ONCE)\n    }\n\n
val mark = markNow()\n    val result = block()\n    return TimedValue(result, mark.elapsedNow())\n}\n"/\n *
Copyright 2010-2020 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is
governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n *\n\npackage
kotlin\n\nimport kotlin.coroutines.*\nimport kotlin.coroutines.intrinsics.*\nimport
kotlin.native.concurrent.SharedImmutable\n\n/**\n * Defines deep recursive function that keeps its stack on the
heap,\n * which allows very deep recursive computations that do not use the actual call stack.\n * To initiate a call to
this deep recursive function use its [invoke] function.\n * As a rule of thumb, it should be used if recursion goes
deeper than a thousand calls.\n *\n * The [DeepRecursiveFunction] takes one parameter of type [T] and returns a
result of type [R].\n * The [block] of code defines the body of a recursive function. In this block\n *
[callRecursive][DeepRecursiveScope.callRecursive] function can be used to make a recursive call\n * to the
declared function. Other instances of [DeepRecursiveFunction] can be called\n * in this scope with `callRecursive`
extension, too.\n *\n * For example, take a look at the following recursive tree class and a deeply\n * recursive
instance of this tree with 100K nodes:\n *\n * ```\n * class Tree(val left: Tree? = null, val right: Tree? = null)\n * val
deepTree = generateSequence(Tree()) { Tree(it) }.take(100_000).last()\n * ```\n *\n * A regular recursive function
can be defined to compute a depth of a tree:\n *\n * ```\n * fun depth(t: Tree?): Int =\n *     if (t == null) 0 else
max(depth(t.left), depth(t.right)) + 1\n * println(depth(deepTree)) // StackOverflowError\n * ```\n *\n * If this
`depth` function is called for a `deepTree` it produces [StackOverflowError] because of deep recursion.\n *
However, the `depth` function can be rewritten using `DeepRecursiveFunction` in the following way, and then\n * it
successfully computes [depth(deepTree)][DeepRecursiveFunction.invoke] expression:\n *\n * ```\n * val depth =
DeepRecursiveFunction<Tree?, Int> { t ->\n *     if (t == null) 0 else max(callRecursive(t.left),
callRecursive(t.right)) + 1\n * }\n * println(depth(deepTree)) // Ok\n * ```\n *\n * Deep recursive functions can also
mutually call each other using a heap for the stack via\n * [callRecursive][DeepRecursiveScope.callRecursive]
extension. For example, the\n * following pair of mutually recursive functions computes the number of tree nodes at
even depth in the tree.\n *\n * ```\n * val mutualRecursion = object {\n *     val even:
DeepRecursiveFunction<Tree?, Int> = DeepRecursiveFunction { t ->\n *         if (t == null) 0 else
odd.callRecursive(t.left) + odd.callRecursive(t.right) + 1\n *     }\n *     val odd: DeepRecursiveFunction<Tree?,
Int> = DeepRecursiveFunction { t ->\n *         if (t == null) 0 else even.callRecursive(t.left) +
even.callRecursive(t.right)\n *     }\n * }\n * ```\n *\n * @param [T] the function parameter type.\n * @param [R]
the function result type.\n * @param block the function body.\n
*\n@SinceKotlin("1.4")\n@ExperimentalStdlibApi\npublic class DeepRecursiveFunction<T, R>(\n    internal val
block: suspend DeepRecursiveScope<T, R>.(T) -> R)\n\n/**\n * Initiates a call to this deep recursive function,
forming a root of the call tree.\n *\n * This operator should not be used from inside of [DeepRecursiveScope] as it
uses the call stack slot for\n * initial recursive invocation. From inside of [DeepRecursiveScope] use\n *
[callRecursive][DeepRecursiveScope.callRecursive].\n
*\n@SinceKotlin("1.4")\n@ExperimentalStdlibApi\npublic operator fun <T, R> DeepRecursiveFunction<T,
R>.invoke(value: T): R =\n    DeepRecursiveScopeImpl<T, R>(block, value).runCallLoop()\n\n/**\n * A scope
class for [DeepRecursiveFunction] function declaration that defines [callRecursive] methods to\n * recursively call
this function or another [DeepRecursiveFunction] putting the call activation frame on the heap.\n *\n * @param [T]

```

```

function parameter type.\n * @param [R] function result type.\n
*\n@RestrictsSuspension\n@SinceKotlin("1.4")\n@ExperimentalStdlibApi\npublic sealed class
DeepRecursiveScope<T, R> {\n    /**\n     * Makes recursive call to this [DeepRecursiveFunction] function putting
the call activation frame on the heap,\n     * as opposed to the actual call stack that is used by a regular recursive
call.\n     */\n    public abstract suspend fun callRecursive(value: T): R\n\n    /**\n     * Makes call to the specified
[DeepRecursiveFunction] function putting the call activation frame on the heap,\n     * as opposed to the actual call
stack that is used by a regular call.\n     */\n    public abstract suspend fun <U, S> DeepRecursiveFunction<U,
S>.callRecursive(value: U): S\n\n    @Deprecated(\n        level = DeprecationLevel.ERROR,\n        message =\n        \"'invoke' should not be called from DeepRecursiveScope. \" +\n        \"Use 'callRecursive' to do recursion in
the heap instead of the call stack.\\\", \n        replaceWith = ReplaceWith(\"this.callRecursive(value)\")\n    )\n    @Suppress(\"UNUSED_PARAMETER\")\n    public operator fun DeepRecursiveFunction<*, *>.invoke(value:
Any?): Nothing =\n        throw UnsupportedOperationException(\"Should not be called from
DeepRecursiveScope\")\n}\n\n// ===== Implementation
=====
\n\n@ExperimentalStdlibApi\nprivate typealias DeepRecursiveFunctionBlock = suspend
DeepRecursiveScope<*, *>.(Any?) -> Any?\n\n@SharedImmutable\nprivate val UNDEFINED_RESULT =
Result.success(COROUTINE_SUSPENDED)\n\n@Suppress(\"UNCHECKED_CAST\")\n@ExperimentalStdlibAp
i\nprivate class DeepRecursiveScopeImpl<T, R>(\n    block: suspend DeepRecursiveScope<T, R>.(T) -> R,\n    value: T\n) : DeepRecursiveScope<T, R>(), Continuation<R> {\n    // Active function block\n    private var
function: DeepRecursiveFunctionBlock = block as DeepRecursiveFunctionBlock\n\n    // Value to call function
with\n    private var value: Any? = value\n\n    // Continuation of the current call\n    private var cont:
Continuation<Any?>? = this as Continuation<Any?>\n\n    // Completion result (completion of the whole call
stack)\n    private var result: Result<Any?> = UNDEFINED_RESULT\n\n    override val context:
CoroutineContext\n        get() = EmptyCoroutineContext\n\n    override fun resumeWith(result: Result<R>) {\n
this.cont = null\n        this.result = result\n    }\n\n    override suspend fun callRecursive(value: T): R =
suspendCoroutineUninterceptedOrReturn { cont ->\n        // calling the same function that is currently active\n
this.cont = cont as Continuation<Any?>\n        this.value = value\n        COROUTINE_SUSPENDED\n    }\n\n    override suspend fun <U, S> DeepRecursiveFunction<U, S>.callRecursive(value: U): S =
suspendCoroutineUninterceptedOrReturn { cont ->\n        // calling another recursive function\n        val function =
block as DeepRecursiveFunctionBlock\n        with(this@DeepRecursiveScopeImpl) {\n            val currentFunction
= this.function\n            if (function !== currentFunction) {\n                // calling a different function -- create a
trampoline to restore function ref\n                this.function = function\n                this.cont =
crossFunctionCompletion(currentFunction, cont as Continuation<Any?>)\n            } else {\n                // calling the
same function -- direct\n                this.cont = cont as Continuation<Any?>\n            }\n            this.value = value\n
        }\n        COROUTINE_SUSPENDED\n    }\n\n    private fun crossFunctionCompletion(\n        currentFunction:
DeepRecursiveFunctionBlock,\n        cont: Continuation<Any?>\n    ): Continuation<Any?> =
Continuation(EmptyCoroutineContext) {\n        this.function = currentFunction\n        // When going back from a
trampoline we cannot just call cont.resume (stack usage!)\n        // We delegate the cont.resumeWith(it) call to
runCallLoop\n        this.cont = cont\n        this.result = it\n    }\n\n    @Suppress(\"UNCHECKED_CAST\")\n    fun
runCallLoop(): R {\n        while (true) {\n            // Note: cont is set to null in DeepRecursiveScopeImpl.resumeWith
when the whole computation completes\n            val result = this.result\n            val cont = this.cont\n            ?:
return (result as Result<R>).getOrThrow() // done -- final result\n            // The order of comparison is important
here for that case of rogue class with broken equals\n            if (UNDEFINED_RESULT == result) {\n                //
call \"function\" with \"value\" using \"cont\" as completion\n                val r = try {\n                    // This is
block.startCoroutine(this, value, cont)\n                    function.startCoroutineUninterceptedOrReturn(this, value,
cont)\n                } catch (e: Throwable) {\n                    cont.resumeWithException(e)\n                    continue\n
                }\n                // If the function returns without suspension -- calls its continuation immediately\n                if (r !==
COROUTINE_SUSPENDED)\n                    cont.resume(r as R)\n            } else {\n                // we returned from a
crossFunctionCompletion trampoline -- call resume here\n                this.result = UNDEFINED_RESULT // reset

```

```

result back\n          cont.resumeWith(result)\n          }\n          }\n          }\n", "/*\n * Copyright 2010-2021
JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the
Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n// Auto-generated file. DO NOT
EDIT!\n\n@file:kotlin.jvm.JvmName("NumbersKt")\n@file:kotlin.jvm.JvmMultifileClass\npackage
kotlin\n\nimport kotlin.math.sign\n\n/** Divides this value by the other value, flooring the result to an integer that is
closer to negative infinity. *\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun
Byte.floorDiv(other: Byte): Int = \n    this.toInt().floorDiv(other.toInt())\n\n/**\n * Calculates the remainder of
flooring division of this value by the other value.\n * \n * The result is either zero or has the same sign as the
_divisor_ and has the absolute value less than the absolute value of the divisor.\n
*\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun Byte.mod(other: Byte): Byte = \n
this.toInt().mod(other.toInt()).toByte()\n\n/** Divides this value by the other value, flooring the result to an integer
that is closer to negative infinity. *\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun
Byte.floorDiv(other: Short): Int = \n    this.toInt().floorDiv(other.toInt())\n\n/**\n * Calculates the remainder of
flooring division of this value by the other value.\n * \n * The result is either zero or has the same sign as the
_divisor_ and has the absolute value less than the absolute value of the divisor.\n
*\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun Byte.mod(other: Short): Short = \n
this.toInt().mod(other.toInt()).toShort()\n\n/** Divides this value by the other value, flooring the result to an integer
that is closer to negative infinity. *\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun
Byte.floorDiv(other: Int): Int = \n    this.toInt().floorDiv(other)\n\n/**\n * Calculates the remainder of flooring
division of this value by the other value.\n * \n * The result is either zero or has the same sign as the _divisor_ and
has the absolute value less than the absolute value of the divisor.\n
*\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun Byte.mod(other: Int): Int = \n
this.toInt().mod(other)\n\n/** Divides this value by the other value, flooring the result to an integer that is closer to
negative infinity. *\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun Byte.floorDiv(other:
Long): Long = \n    this.toLong().floorDiv(other)\n\n/**\n * Calculates the remainder of flooring division of this
value by the other value.\n * \n * The result is either zero or has the same sign as the _divisor_ and has the absolute
value less than the absolute value of the divisor.\n *\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic
inline fun Byte.mod(other: Long): Long = \n    this.toLong().mod(other)\n\n/** Divides this value by the other
value, flooring the result to an integer that is closer to negative infinity.
*\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun Short.floorDiv(other: Byte): Int = \n
this.toInt().floorDiv(other.toInt())\n\n/**\n * Calculates the remainder of flooring division of this value by the other
value.\n * \n * The result is either zero or has the same sign as the _divisor_ and has the absolute value less than the
absolute value of the divisor.\n *\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun
Short.mod(other: Byte): Byte = \n    this.toInt().mod(other.toInt()).toByte()\n\n/** Divides this value by the other
value, flooring the result to an integer that is closer to negative infinity.
*\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun Short.floorDiv(other: Short): Int = \n
this.toInt().floorDiv(other.toInt())\n\n/**\n * Calculates the remainder of flooring division of this value by the other
value.\n * \n * The result is either zero or has the same sign as the _divisor_ and has the absolute value less than the
absolute value of the divisor.\n *\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun
Short.mod(other: Short): Short = \n    this.toInt().mod(other.toInt()).toShort()\n\n/** Divides this value by the other
value, flooring the result to an integer that is closer to negative infinity.
*\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun Short.floorDiv(other: Int): Int = \n
this.toInt().floorDiv(other)\n\n/**\n * Calculates the remainder of flooring division of this value by the other
value.\n * \n * The result is either zero or has the same sign as the _divisor_ and has the absolute value less than the
absolute value of the divisor.\n *\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun
Short.mod(other: Int): Int = \n    this.toInt().mod(other)\n\n/** Divides this value by the other value, flooring the
result to an integer that is closer to negative infinity.
*\n\n@SinceKotlin("1.5")\n@kotlin.internal.InlineOnly\npublic inline fun Short.floorDiv(other: Long): Long = \n

```

`this.toLong().floorDiv(other)` Calculates the remainder of flooring division of this value by the other value. The result is either zero or has the same sign as the `_divisor_` and has the absolute value less than the absolute value of the divisor.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Short.mod(other: Long): Long = \n    this.toLong().mod(other)
```

 Divides this value by the other value, flooring the result to an integer that is closer to negative infinity.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Int.floorDiv(other: Byte): Int = \n    this.floorDiv(other.toInt())
```

 Calculates the remainder of flooring division of this value by the other value. The result is either zero or has the same sign as the `_divisor_` and has the absolute value less than the absolute value of the divisor.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Int.mod(other: Byte): Byte = \n    this.mod(other.toInt()).toByte()
```

 Divides this value by the other value, flooring the result to an integer that is closer to negative infinity.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Int.floorDiv(other: Short): Int = \n    this.floorDiv(other.toInt())
```

 Calculates the remainder of flooring division of this value by the other value. The result is either zero or has the same sign as the `_divisor_` and has the absolute value less than the absolute value of the divisor.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Int.mod(other: Short): Short = \n    this.mod(other.toInt()).toShort()
```

 Divides this value by the other value, flooring the result to an integer that is closer to negative infinity.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Int.floorDiv(other: Int): Int {\n    var q = this / other\n    if (this xor other < 0 && q * other != this) q--\n    return q}
```

 Calculates the remainder of flooring division of this value by the other value. The result is either zero or has the same sign as the `_divisor_` and has the absolute value less than the absolute value of the divisor.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Int.mod(other: Int): Int {\n    val r = this % other\n    return r + (other and (((r xor other) and (r or -r)) shr 31))}
```

 Divides this value by the other value, flooring the result to an integer that is closer to negative infinity.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Int.floorDiv(other: Long): Long = \n    this.toLong().floorDiv(other)
```

 Calculates the remainder of flooring division of this value by the other value. The result is either zero or has the same sign as the `_divisor_` and has the absolute value less than the absolute value of the divisor.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Int.mod(other: Long): Long = \n    this.toLong().mod(other)
```

 Divides this value by the other value, flooring the result to an integer that is closer to negative infinity.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Long.floorDiv(other: Byte): Long = \n    this.floorDiv(other.toLong())
```

 Calculates the remainder of flooring division of this value by the other value. The result is either zero or has the same sign as the `_divisor_` and has the absolute value less than the absolute value of the divisor.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Long.mod(other: Byte): Byte = \n    this.mod(other.toLong()).toByte()
```

 Divides this value by the other value, flooring the result to an integer that is closer to negative infinity.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Long.floorDiv(other: Short): Long = \n    this.floorDiv(other.toLong())
```

 Calculates the remainder of flooring division of this value by the other value. The result is either zero or has the same sign as the `_divisor_` and has the absolute value less than the absolute value of the divisor.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Long.mod(other: Short): Short = \n    this.mod(other.toLong()).toShort()
```

 Divides this value by the other value, flooring the result to an integer that is closer to negative infinity.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Long.floorDiv(other: Int): Long = \n    this.floorDiv(other.toLong())
```

 Calculates the remainder of flooring division of this value by the other value. The result is either zero or has the same sign as the `_divisor_` and has the absolute value less than the absolute value of the divisor.

```
@SinceKotlin("1.5")@kotlin.internal.InlineOnly\npublic inline fun Long.mod(other: Int): Int = \n    this.mod(other.toLong()).toInt()
```

 Divides this value by the other value, flooring the result to an integer that is closer to negative infinity.



```

version is not less than the version specified\n    * with the provided [major] and [minor] components.\n    */\n    public fun isAtLeast(major: Int, minor: Int): Boolean = // this.version >= versionOf(major, minor, 0)\n    this.major > major || (this.major == major &&\n        this.minor >= minor)\n    /**\n     * Returns `true` if this
version is not less than the version specified\n    * with the provided [major], [minor] and [patch] components.\n    */\n    public fun isAtLeast(major: Int, minor: Int, patch: Int): Boolean = // this.version >= versionOf(major, minor,\n    patch)\n    this.major > major || (this.major == major &&\n        (this.minor > minor || this.minor == minor\n    &&\n        this.patch >= patch))\n    companion object {\n        /**\n         * Maximum value a version
component can have, a constant value 255.\n        */\n        // NOTE: Must be placed before CURRENT because its
initialization requires this field being initialized in JS\n        public const val MAX_COMPONENT_VALUE =
255\n        /**\n         * Returns the current version of the Kotlin standard library.\n        */\n        @kotlin.jvm.JvmField\n        public val CURRENT: KotlinVersion = KotlinVersionCurrentValue.get()\n    }\n}\n\n// this class is ignored during classpath normalization when considering whether to recompile dependencies
in Kotlin build\nprivate object KotlinVersionCurrentValue {\n    @kotlin.jvm.JvmStatic\n    fun get():\n    KotlinVersion = KotlinVersion(1, 6, 10) // value is written here automatically during build\n    },"/*\n    * Copyright
2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n    * Use of this source code is governed
by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*/\n\n@file:kotlin.jvm.JvmName("LateinitKt")\n@file:Suppress("unused")\n\npackage kotlin\n\nimport
kotlin.internal.InlineOnly\nimport kotlin.internal.AccessibleLateinitPropertyLiteral\nimport
kotlin.reflect.KProperty0\n\n/**\n * Returns `true` if this lateinit property has been assigned a value, and `false`
otherwise.\n * Cannot be used in an inline function, to avoid binary compatibility issues.\n
*/\n\n@SinceKotlin("1.2")\n@InlineOnly\ninline val @receiver:AccessibleLateinitPropertyLiteral
KProperty0<*>.isInitialized: Boolean\n    get() = throw NotImplementedError("Implementation is
intrinsic")\n    },"/*\n    * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n    * Use
of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n
*/\n\n@file:kotlin.jvm.JvmName("LazyKt")\n@file:kotlin.jvm.JvmMultifileClass\n\npackage kotlin\n\nimport
kotlin.reflect.KProperty\n\n/**\n * Represents a value with lazy initialization.\n * To create an instance of
[Lazy] use the [lazy] function.\n */\n\npublic interface Lazy<out T> {\n    /**\n     * Gets the lazily initialized value of
the current Lazy instance.\n     * Once the value was initialized it must not change during the rest of lifetime of this
Lazy instance.\n     */\n    public val value: T\n    /**\n     * Returns `true` if a value for this Lazy instance has been
already initialized, and `false` otherwise.\n     * Once this function has returned `true` it stays `true` for the rest of
lifetime of this Lazy instance.\n     */\n    public fun isInitialized(): Boolean\n}\n\n/**\n * Creates a new instance of
the [Lazy] that is already initialized with the specified [value].\n */\n\npublic fun <T> lazyOf(value: T): Lazy<T> =
InitializedLazyImpl(value)\n\n/**\n * An extension to delegate a read-only property of type [T] to an instance of
[Lazy].\n * This extension allows to use instances of Lazy for property delegation:\n * `val property: String by
lazy { initializer }`\n */\n\n@kotlin.internal.InlineOnly\npublic inline operator fun <T> Lazy<T>.getValue(thisRef:
Any?, property: KProperty<*>): T = value\n\n/**\n * Specifies how a [Lazy] instance synchronizes initialization
among multiple threads.\n */\n\npublic enum class LazyThreadSafetyMode {\n    /**\n     * Locks are used to ensure
that only a single thread can initialize the [Lazy] instance.\n     */\n    SYNCHRONIZED,\n    /**\n     * Initializer
function can be called several times on concurrent access to uninitialized [Lazy] instance value,\n     * but only the
first returned value will be used as the value of [Lazy] instance.\n     */\n    PUBLICATION,\n    /**\n     * No
locks are used to synchronize an access to the [Lazy] instance value; if the instance is accessed from multiple
threads, its behavior is undefined.\n     */\n    NONE\n}\n\ninternal object
UNINITIALIZED_VALUE\n\n// internal to be called from lazy in JS\ninternal class UnsafeLazyImpl<out
T>(initializer: () -> T) : Lazy<T>, Serializable {\n    private var initializer: (() -> T)? = initializer\n    private var
_value: Any? = UNINITIALIZED_VALUE\n\n    override val value: T\n        get() {\n            if (_value ===
UNINITIALIZED_VALUE) {\n                _value = initializer!!()\n                initializer = null\n            }\n
        }\n\n    @Suppress("UNCHECKED_CAST")\n        return _value as T\n    }\n\n    override fun isInitialized():

```

```

Boolean = _value !== UNINITIALIZED_VALUE\n\n  override fun toString(): String = if (isInitialized())
value.toString() else \"Lazy value not initialized yet.\"\n\n  private fun writeReplace(): Any =
InitializedLazyImpl(value)\n}\n\ninternal class InitializedLazyImpl<out T>(override val value: T) : Lazy<T>,
Serializable {\n\n  override fun isInitialized(): Boolean = true\n\n  override fun toString(): String =
value.toString()\n}\n}\n\", /*\n * Copyright 2010-2019 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n
*\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName(\"NumbersKt\")\npackage kotlin\n\n/**\n *
Counts the number of set bits in the binary representation of this [Int] number.\n
*\n\n@SinceKotlin(\"1.4\")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Int.countOneBits(): Int\n\n/**\n * Counts the number of consecutive most significant bits that are zero in the binary
representation of this [Int] number.\n
*\n\n@SinceKotlin(\"1.4\")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Int.countLeadingZeroBits(): Int\n\n/**\n * Counts the number of consecutive least significant bits that are zero in
the binary representation of this [Int] number.\n
*\n\n@SinceKotlin(\"1.4\")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Int.countTrailingZeroBits(): Int\n\n/**\n * Returns a number having a single bit set in the position of the most
significant set bit of this [Int] number,\n * or zero, if this number is zero.\n
*\n\n@SinceKotlin(\"1.4\")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Int.takeHighestOneBit(): Int\n\n/**\n * Returns a number having a single bit set in the position of the least
significant set bit of this [Int] number,\n * or zero, if this number is zero.\n
*\n\n@SinceKotlin(\"1.4\")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Int.takeLowestOneBit(): Int\n\n/**\n * Rotates the binary representation of this [Int] number left by the specified
[bitCount] number of bits.\n * The most significant bits pushed out from the left side reenter the number as the least
significant bits on the right side.\n * Rotating the number left by a negative bit count is the same as rotating it
right by the negated bit count:\n * `number.rotateLeft(-n) == number.rotateRight(n)`\n * Rotating by a multiple
of [Int.SIZE_BITS] (32) returns the same number, or more generally\n * `number.rotateLeft(n) ==
number.rotateLeft(n % 32)`\n
*\n\n@SinceKotlin(\"1.6\")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Int.rotateLeft(bitCount: Int): Int\n\n/**\n * Rotates the binary representation of this [Int] number right by the
specified [bitCount] number of bits.\n * The least significant bits pushed out from the right side reenter the number
as the most significant bits on the left side.\n * Rotating the number right by a negative bit count is the same as
rotating it left by the negated bit count:\n * `number.rotateRight(-n) == number.rotateLeft(n)`\n * Rotating by a
multiple of [Int.SIZE_BITS] (32) returns the same number, or more generally\n * `number.rotateRight(n) ==
number.rotateRight(n % 32)`\n
*\n\n@SinceKotlin(\"1.6\")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Int.rotateRight(bitCount: Int): Int\n\n/**\n * Counts the number of set bits in the binary representation of this
[Long] number.\n
*\n\n@SinceKotlin(\"1.4\")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect
fun Long.countOneBits(): Int\n\n/**\n * Counts the number of consecutive most significant bits that are zero in the
binary representation of this [Long] number.\n
*\n\n@SinceKotlin(\"1.4\")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Long.countLeadingZeroBits(): Int\n\n/**\n * Counts the number of consecutive least significant bits that are zero in
the binary representation of this [Long] number.\n
*\n\n@SinceKotlin(\"1.4\")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Long.countTrailingZeroBits(): Int\n\n/**\n * Returns a number having a single bit set in the position of the most
significant set bit of this [Long] number,\n * or zero, if this number is zero.\n
*\n\n@SinceKotlin(\"1.4\")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Long.takeHighestOneBit(): Long\n\n/**\n * Returns a number having a single bit set in the position of the least

```

significant set bit of this [Long] number, \n \* or zero, if this number is zero.\n

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Long.takeLowestOneBit(): Long\n\n/**\n * Rotates the binary representation of this [Long] number left by the
specified [bitCount] number of bits.\n * The most significant bits pushed out from the left side reenter the number as
the least significant bits on the right side.\n *\n * Rotating the number left by a negative bit count is the same as
rotating it right by the negated bit count:\n * `number.rotateLeft(-n) == number.rotateRight(n)`\n *\n * Rotating by a
multiple of [Long.SIZE_BITS] (64) returns the same number, or more generally\n * `number.rotateLeft(n) ==
number.rotateLeft(n % 64)`\n
```

```
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Long.rotateLeft(bitCount: Int): Long\n\n/**\n * Rotates the binary representation of this [Long] number right by the
specified [bitCount] number of bits.\n * The least significant bits pushed out from the right side reenter the number
as the most significant bits on the left side.\n *\n * Rotating the number right by a negative bit count is the same as
rotating it left by the negated bit count:\n * `number.rotateRight(-n) == number.rotateLeft(n)`\n *\n * Rotating by a
multiple of [Long.SIZE_BITS] (64) returns the same number, or more generally\n * `number.rotateRight(n) ==
number.rotateRight(n % 64)`\n
```

```
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic expect fun
Long.rotateRight(bitCount: Int): Long\n\n/**\n * Counts the number of set bits in the binary representation of this
[Byte] number.\n
```

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun Byte.countOneBits(): Int = (toInt() and 0xFF).countOneBits()\n\n/**\n * Counts the number of
consecutive most significant bits that are zero in the binary representation of this [Byte] number.\n
```

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun Byte.countLeadingZeroBits(): Int = (toInt() and 0xFF).countLeadingZeroBits() - (Int.SIZE_BITS -
Byte.SIZE_BITS)\n\n/**\n * Counts the number of consecutive least significant bits that are zero in the binary
representation of this [Byte] number.\n
```

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun Byte.countTrailingZeroBits(): Int = (toInt() or 0x100).countTrailingZeroBits()\n\n/**\n * Returns a
number having a single bit set in the position of the most significant set bit of this [Byte] number, \n * or zero, if this
number is zero.\n
```

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun Byte.takeHighestOneBit(): Byte = (toInt() and 0xFF).takeHighestOneBit().toByte()\n\n/**\n * Returns a
number having a single bit set in the position of the least significant set bit of this [Byte] number, \n * or zero, if this
number is zero.\n
```

```
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun Byte.takeLowestOneBit(): Byte = toInt().takeLowestOneBit().toByte()\n\n/**\n * Rotates the binary
representation of this [Byte] number left by the specified [bitCount] number of bits.\n * The most significant bits
pushed out from the left side reenter the number as the least significant bits on the right side.\n *\n * Rotating the
number left by a negative bit count is the same as rotating it right by the negated bit count:\n * `number.rotateLeft(-
n) == number.rotateRight(n)`\n *\n * Rotating by a multiple of [Byte.SIZE_BITS] (8) returns the same number, or
more generally\n * `number.rotateLeft(n) == number.rotateLeft(n % 8)`\n
```

```
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
Byte.rotateLeft(bitCount: Int): Byte =\n    (toInt().shl(bitCount and 7) or (toInt() and 0xFF).ushr(8 - (bitCount and
7))).toByte()\n\n/**\n * Rotates the binary representation of this [Byte] number right by the specified [bitCount]
number of bits.\n * The least significant bits pushed out from the right side reenter the number as the most
significant bits on the left side.\n *\n * Rotating the number right by a negative bit count is the same as rotating it
left by the negated bit count:\n * `number.rotateRight(-n) == number.rotateLeft(n)`\n *\n * Rotating by a multiple of
[Byte.SIZE_BITS] (8) returns the same number, or more generally\n * `number.rotateRight(n) ==
number.rotateRight(n % 8)`\n
```

```

*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
Byte.rotateRight(bitCount: Int): Byte =\n (toInt().shl(8 - (bitCount and 7)) or (toInt() and 0xFF).ushr(bitCount and
7)).toByte()\n\n**\n * Counts the number of set bits in the binary representation of this [Short] number.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun Short.countOneBits(): Int = (toInt() and 0xFFFF).countOneBits()\n\n**\n * Counts the number of
consecutive most significant bits that are zero in the binary representation of this [Short] number.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun Short.countLeadingZeroBits(): Int =\n (toInt() and 0xFFFF).countLeadingZeroBits() - (Int.SIZE_BITS
- Short.SIZE_BITS)\n\n**\n * Counts the number of consecutive least significant bits that are zero in the binary
representation of this [Short] number.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun Short.countTrailingZeroBits(): Int = (toInt() or 0x10000).countTrailingZeroBits()\n\n**\n * Returns a
number having a single bit set in the position of the most significant set bit of this [Short] number,\n * or zero, if this
number is zero.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun Short.takeHighestOneBit(): Short = (toInt() and 0xFFFF).takeHighestOneBit().toShort()\n\n**\n *
Returns a number having a single bit set in the position of the least significant set bit of this [Short] number,\n * or
zero, if this number is zero.\n
*\n@SinceKotlin("1.4")\n@WasExperimental(ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic
inline fun Short.takeLowestOneBit(): Short = toInt().takeLowestOneBit().toShort()\n\n**\n * Rotates the binary
representation of this [Short] number left by the specified [bitCount] number of bits.\n * The most significant bits
pushed out from the left side reenter the number as the least significant bits on the right side.\n * Rotating the
number left by a negative bit count is the same as rotating it right by the negated bit count:\n * `number.rotateLeft(-
n) == number.rotateRight(n)`\n * Rotating by a multiple of [Short.SIZE_BITS] (16) returns the same number, or
more generally\n * `number.rotateLeft(n) == number.rotateLeft(n % 16)`\n
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
Short.rotateLeft(bitCount: Int): Short =\n (toInt().shl(bitCount and 15) or (toInt() and 0xFFFF).ushr(16 - (bitCount
and 15))).toShort()\n\n**\n * Rotates the binary representation of this [Short] number right by the specified
[bitCount] number of bits.\n * The least significant bits pushed out from the right side reenter the number as the
most significant bits on the left side.\n * Rotating the number right by a negative bit count is the same as rotating
it left by the negated bit count:\n * `number.rotateRight(-n) == number.rotateLeft(n)`\n * Rotating by a multiple
of [Short.SIZE_BITS] (16) returns the same number, or more generally\n * `number.rotateRight(n) ==
number.rotateRight(n % 16)`\n
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class)\npublic fun
Short.rotateRight(bitCount: Int): Short =\n (toInt().shl(16 - (bitCount and 15)) or (toInt() and
0xFFFF).ushr(bitCount and 15)).toShort()\n\n**\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming
Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n\npackage kotlin\nimport kotlin.internal.RequireKotlin\nimport
kotlin.internal.RequireKotlinVersionKind\n\n@kotlin.internal.InlineOnly\n@SinceKotlin("1.2")\n@Suppress("IN
VISIBLE_MEMBER", "INVISIBLE_REFERENCE")\n@RequireKotlin("1.2.30", level =
DeprecationLevel.HIDDEN, versionKind = RequireKotlinVersionKind.COMPILER_VERSION)\npublic inline fun
<R> suspend(noinline block: suspend () -> R): suspend () -> R = block\n\n**\n * Copyright 2010-2018 JetBrains
s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0
license that can be found in the license/LICENSE.txt file.\n
*\n\n@file:kotlin.jvm.JvmName("TuplesKt")\n\npackage kotlin\n\n**\n * Represents a generic pair of two
values.\n * There is no meaning attached to values in this class, it can be used for any purpose.\n * Pair exhibits
value semantics, i.e. two pairs are equal if both components are equal.\n * An example of decomposing it into
values:\n * @sample samples.misc.Tuples.pairDestructuring\n * @param A type of the first value.\n * @param

```

```

B type of the second value.
 * @property first First value.
 * @property second Second value.
 * @constructor
Creates a new instance of Pair.
 ^\npublic data class Pair<out A, out B>(\n  public val first: A,\n  public val
second: B)\n) : Serializable {\n\n  /**\n   * Returns string representation of the [Pair] including its [first] and
[second] values.\n   */\n  public override fun toString(): String = \"($first, $second)\"\n}\n\n/**\n * Creates a tuple
of type [Pair] from this and [that].\n */\n * This can be useful for creating [Map] literals with less noise, for
example:\n * @sample samples.collections.Maps.Instantiation.mapFromPairs\n */\npublic infix fun <A, B>
A.to(that: B): Pair<A, B> = Pair(this, that)\n\n/**\n * Converts this pair into a list.\n * @sample
samples.misc.Tuples.pairToList\n */\npublic fun <T> Pair<T, T>.toList(): List<T> = listOf(first, second)\n\n/**\n *
Represents a triad of values\n */\n * There is no meaning attached to values in this class, it can be used for any
purpose.\n * Triple exhibits value semantics, i.e. two triples are equal if all three components are equal.\n * An
example of decomposing it into values:\n * @sample samples.misc.Tuples.tripleDestructuring\n */\n * @param A
type of the first value.\n * @param B type of the second value.\n * @param C type of the third value.\n * @property
first First value.\n * @property second Second value.\n * @property third Third value.\n */\npublic data class
Triple<out A, out B, out C>(\n  public val first: A,\n  public val second: B,\n  public val third: C)\n) : Serializable
{\n\n  /**\n   * Returns string representation of the [Triple] including its [first], [second] and [third] values.\n   */\n
  public override fun toString(): String = \"($first, $second, $third)\"\n}\n\n/**\n * Converts this triple into a
list.\n * @sample samples.misc.Tuples.tripleToList\n */\npublic fun <T> Triple<T, T, T>.toList(): List<T> =
listOf(first, second, third)\n\n", /*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language
contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the
license/LICENSE.txt file.\n */\n\n// Auto-generated file. DO NOT EDIT!\n\npackage kotlin.ranges\n\n\nimport
kotlin.internal.*\n\n/**\n * A range of values of type `UInt`.\n */\n\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic class UIntRange(start:
UInt, endInclusive: UInt) : UIntProgression(start, endInclusive, 1), ClosedRange<UInt> {\n  override val start:
UInt get() = first\n  override val endInclusive: UInt get() = last\n\n  override fun contains(value: UInt): Boolean =
first <= value && value <= last\n\n  /**\n   * Checks if the range is empty.\n   */\n   * The range is empty if its
start value is greater than the end value.\n   */\n  override fun isEmpty(): Boolean = first > last\n\n  override fun
equals(other: Any?): Boolean =\n    other is UIntRange && (isEmpty() && other.isEmpty()) ||\n    first ==
other.first && last == other.last\n\n  override fun hashCode(): Int =\n    if (isEmpty()) -1 else (31 * first.toInt()
+ last.toInt())\n\n  override fun toString(): String = \"$first..$last\"\n\n  companion object {\n    /** An empty
range of values of type UInt. */\n    public val EMPTY: UIntRange = UIntRange(UInt.MAX_VALUE,
UInt.MIN_VALUE)\n  }\n}\n\n/**\n * A progression of values of type `UInt`.\n */\n\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic open class
UIntProgression\ninternal constructor(\n  start: UInt,\n  endInclusive: UInt,\n  step: Int)\n: Iterable<UInt> {\n  init {\n    if (step == 0.toInt()) throw kotlin.IllegalArgumentException(\"Step must be non-zero.\")\n    if (step
== Int.MIN_VALUE) throw kotlin.IllegalArgumentException(\"Step must be greater than Int.MIN_VALUE to
avoid overflow on negation.\")\n  }\n\n  /**\n   * The first element in the progression.\n   */\n  public val first:
UInt = start\n\n  /**\n   * The last element in the progression.\n   */\n  public val last: UInt =
getProgressionLastElement(start, endInclusive, step)\n\n  /**\n   * The step of the progression.\n   */\n  public
val step: Int = step\n\n  final override fun iterator(): Iterator<UInt> = UIntProgressionIterator(first, last, step)\n\n  /**\n   * Checks if the progression is empty.\n   */\n   * Progression with a positive step is empty if its first
element is greater than the last element.\n   * Progression with a negative step is empty if its first element is less
than the last element.\n   */\n  public open fun isEmpty(): Boolean = if (step > 0) first > last else first < last\n\n
override fun equals(other: Any?): Boolean =\n    other is UIntProgression && (isEmpty() && other.isEmpty()) ||\n    first ==
other.first && last == other.last && step == other.step)\n\n  override fun hashCode(): Int =\n    if (isEmpty()) -1 else (31 * (31 * first.toInt() + last.toInt()) + step.toInt())\n\n  override fun toString(): String = if
(step > 0) \"$first..$last step $step\" else \"$first downTo $last step ${-step}\"\n\n  companion object {\n    /**\n   * Creates UIntProgression within the specified bounds of a closed range.\n   */\n   * The progression starts with
the [rangeStart] value and goes toward the [rangeEnd] value not excluding it, with the specified [step].\n   */\n    In

```



```

progression.\n    *\n    public val first: ULong = start\n    /**\n     * The last element in the progression.\n    *\n    public val last: ULong = getProgressionLastElement(start, endInclusive, step)\n    /**\n     * The step of the progression.\n    *\n    public val step: Long = step\n    final override fun iterator(): Iterator<ULong> = ULongProgressionIterator(first, last, step)\n    /**\n     *\n     * Checks if the progression is empty.\n     *\n     * Progression with a positive step is empty if its first element is greater than the last element.\n     * Progression with a negative step is empty if its first element is less than the last element.\n    *\n    public open fun isEmpty(): Boolean = if (step > 0) first > last else first < last\n    override fun equals(other: Any?): Boolean =\n        other is ULongProgression && (isEmpty() && other.isEmpty()) ||\n            first == other.first && last == other.last && step == other.step\n    override fun hashCode(): Int =\n        if (isEmpty()) -1 else (31 * (31 * (first xor (first shr 32)).toInt() + (last xor (last shr 32)).toInt()) + (step xor (step ushr 32)).toInt())\n    override fun toString(): String = if (step > 0) \"$first..$last step $step\" else \"$first downTo $last step ${-step}\"\n    companion object {\n    /**\n     * Creates ULongProgression within the specified bounds of a closed range.\n     *\n     * The progression starts with the [rangeStart] value and goes toward the [rangeEnd] value not excluding it, with the specified [step].\n     *\n     * In order to go backwards the [step] must be negative.\n     *\n     * [step] must be greater than `Long.MIN_VALUE` and not equal to zero.\n    *\n    public fun fromClosedRange(rangeStart: ULong, rangeEnd: ULong, step: Long): ULongProgression = ULongProgression(rangeStart, rangeEnd, step)\n    }\n    }\n    /**\n     * An iterator over a progression of values of type `ULong`.\n     * @property step the number by which the value is incremented on each step.\n    *\n    *\n    @SinceKotlin("1.3")\n    @Suppress("DEPRECATION_ERROR")\n    private class ULongProgressionIterator(first: ULong, last: ULong, step: Long) : ULongIterator() {\n        private val finalElement = last\n        private var hasNext: Boolean = if (step > 0) first <= last else first >= last\n        private val step = step.toULong() // use 2-complement math for negative steps\n        private var next = if (hasNext) first else finalElement\n        override fun hasNext(): Boolean = hasNext\n        override fun nextULong(): ULong {\n            val value = next\n            if (value == finalElement) {\n                if (!hasNext) throw kotlin.NoSuchElementException()\n                hasNext = false\n            } else {\n                next += step\n            }\n            return value\n        }\n    }\n    "\n    /**\n     * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n     * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n    */\n    package kotlin.math\n    /**\n     *\n     * Returns the smaller of two values.\n    *\n    @SinceKotlin("1.5")\n    @WasExperimental(ExperimentalUnsignedTypes::class)\n    @kotlin.internal.InlineOnly\n    public inline fun min(a: UInt, b: UInt): UInt {\n        return minOf(a, b)\n    }\n    /**\n     *\n     * Returns the smaller of two values.\n    *\n    @SinceKotlin("1.5")\n    @WasExperimental(ExperimentalUnsignedTypes::class)\n    @kotlin.internal.InlineOnly\n    public inline fun min(a: ULong, b: ULong): ULong {\n        return minOf(a, b)\n    }\n    /**\n     *\n     * Returns the greater of two values.\n    *\n    @SinceKotlin("1.5")\n    @WasExperimental(ExperimentalUnsignedTypes::class)\n    @kotlin.internal.InlineOnly\n    public inline fun max(a: UInt, b: UInt): UInt {\n        return maxOf(a, b)\n    }\n    /**\n     *\n     * Returns the greater of two values.\n    *\n    @SinceKotlin("1.5")\n    @WasExperimental(ExperimentalUnsignedTypes::class)\n    @kotlin.internal.InlineOnly\n    public inline fun max(a: ULong, b: ULong): ULong {\n        return maxOf(a, b)\n    }\n    "\n    /**\n     * Copyright 2010-2021 JetBrains s.r.o. and Kotlin Programming Language contributors.\n     * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n    */\n    @file:kotlin.jvm.JvmName("UNumbersKt")\n    package kotlin\n    /**\n     *\n     * Counts the number of set bits in the binary representation of this [UInt] number.\n    *\n    @SinceKotlin("1.5")\n    @WasExperimental(ExperimentalUnsignedTypes::class, ExperimentalStdlibApi::class)\n    @kotlin.internal.InlineOnly\n    public inline fun UInt.countOneBits(): Int = toInt().countOneBits()\n    /**\n     *\n     * Counts the number of consecutive most significant bits that are zero in the binary representation of this [UInt] number.\n    *\n    @SinceKotlin("1.5")\n    @WasExperimental(ExperimentalUnsignedTypes::class,

```

```

ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UInt.countLeadingZeroBits(): Int =
toInt().countLeadingZeroBits()\n\n/**\n * Counts the number of consecutive least significant bits that are zero in the
binary representation of this [UInt] number.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UInt.countTrailingZeroBits(): Int =
toInt().countTrailingZeroBits()\n\n/**\n * Returns a number having a single bit set in the position of the most
significant set bit of this [UInt] number,\n * or zero, if this number is zero.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UInt.takeHighestOneBit(): UInt =
toInt().takeHighestOneBit().toUInt()\n\n/**\n * Returns a number having a single bit set in the position of the least
significant set bit of this [UInt] number,\n * or zero, if this number is zero.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UInt.takeLowestOneBit(): UInt =
toInt().takeLowestOneBit().toUInt()\n\n/**\n * Rotates the binary representation of this [UInt] number left by the
specified [bitCount] number of bits.\n * The most significant bits pushed out from the left side reenter the number as
the least significant bits on the right side.\n * \n * Rotating the number left by a negative bit count is the same as
rotating it right by the negated bit count:\n * `number.rotateLeft(-n) == number.rotateRight(n)`\n * \n * Rotating by a
multiple of [UInt.SIZE_BITS] (32) returns the same number, or more generally\n * `number.rotateLeft(n) ==
number.rotateLeft(n % 32)`\n *\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class,
ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\npublic inline fun UInt.rotateLeft(bitCount: Int):
UInt = toInt().rotateLeft(bitCount).toUInt()\n\n/**\n * Rotates the binary representation of this [UInt] number
right by the specified [bitCount] number of bits.\n * The least significant bits pushed out from the right side reenter
the number as the most significant bits on the left side.\n * \n * Rotating the number right by a negative bit count is
the same as rotating it left by the negated bit count:\n * `number.rotateRight(-n) == number.rotateLeft(n)`\n * \n *
Rotating by a multiple of [UInt.SIZE_BITS] (32) returns the same number, or more generally\n *
`number.rotateRight(n) == number.rotateRight(n % 32)`\n
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class,
ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\npublic inline fun UInt.rotateRight(bitCount: Int):
UInt = toInt().rotateRight(bitCount).toUInt()\n\n/**\n * Counts the number of set bits in the binary representation
of this [ULong] number.\n *\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun ULong.countOneBits(): Int =
toLong().countOneBits()\n\n/**\n * Counts the number of consecutive most significant bits that are zero in the
binary representation of this [ULong] number.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun ULong.countLeadingZeroBits(): Int =
toLong().countLeadingZeroBits()\n\n/**\n * Counts the number of consecutive least significant bits that are zero
in the binary representation of this [ULong] number.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun ULong.countTrailingZeroBits(): Int =
toLong().countTrailingZeroBits()\n\n/**\n * Returns a number having a single bit set in the position of the most
significant set bit of this [ULong] number,\n * or zero, if this number is zero.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun ULong.takeHighestOneBit(): ULong =
toLong().takeHighestOneBit().toULong()\n\n/**\n * Returns a number having a single bit set in the position of the
least significant set bit of this [ULong] number,\n * or zero, if this number is zero.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun ULong.takeLowestOneBit(): ULong =
toLong().takeLowestOneBit().toULong()\n\n/**\n * Rotates the binary representation of this [ULong] number left

```

by the specified [bitCount] number of bits.\n \* The most significant bits pushed out from the left side reenter the number as the least significant bits on the right side.\n \* Rotating the number left by a negative bit count is the same as rotating it right by the negated bit count:\n \* `number.rotateLeft(-n) == number.rotateRight(n)`\n \* Rotating by a multiple of [ULong.SIZE\_BITS] (64) returns the same number, or more generally\n \* `number.rotateLeft(n) == number.rotateLeft(n % 64)`\n

```
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class,  
ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\npublic inline fun ULong.rotateLeft(bitCount:  
Int): ULong = toLong().rotateLeft(bitCount).toULong()\n\n/**\n * Rotates the binary representation of this [ULong]  
number right by the specified [bitCount] number of bits.\n * The least significant bits pushed out from the right side  
reenter the number as the most significant bits on the left side.\n * Rotating the number right by a negative bit  
count is the same as rotating it left by the negated bit count:\n * `number.rotateRight(-n) == number.rotateLeft(n)`\n *\n * Rotating by a multiple of [ULong.SIZE_BITS] (64) returns the same number, or more generally\n *\n * `number.rotateRight(n) == number.rotateRight(n % 64)`\n
```

```
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class,  
ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\npublic inline fun ULong.rotateRight(bitCount:  
Int): ULong = toLong().rotateRight(bitCount).toULong()\n\n/**\n * Counts the number of set bits in the binary  
representation of this [UByte] number.\n
```

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,  
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UByte.countOneBits(): Int =  
toUInt().countOneBits()\n\n/**\n * Counts the number of consecutive most significant bits that are zero in the  
binary representation of this [UByte] number.\n
```

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,  
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UByte.countLeadingZeroBits(): Int =  
toByte().countLeadingZeroBits()\n\n/**\n * Counts the number of consecutive least significant bits that are zero in  
the binary representation of this [UByte] number.\n
```

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,  
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UByte.countTrailingZeroBits(): Int =  
toByte().countTrailingZeroBits()\n\n/**\n * Returns a number having a single bit set in the position of the most  
significant set bit of this [UByte] number,\n * or zero, if this number is zero.\n
```

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,  
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UByte.takeHighestOneBit(): UByte  
= toInt().takeHighestOneBit().toUByte()\n\n/**\n * Returns a number having a single bit set in the position of the  
least significant set bit of this [UByte] number,\n * or zero, if this number is zero.\n
```

```
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,  
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UByte.takeLowestOneBit(): UByte =  
toInt().takeLowestOneBit().toUByte()\n\n/**\n * Rotates the binary representation of this [UByte] number left by  
the specified [bitCount] number of bits.\n * The most significant bits pushed out from the left side reenter the  
number as the least significant bits on the right side.\n * Rotating the number left by a negative bit count is the  
same as rotating it right by the negated bit count:\n * `number.rotateLeft(-n) == number.rotateRight(n)`\n *\n * Rotating by a multiple of [UByte.SIZE_BITS] (8) returns the same number, or more generally\n *\n * `number.rotateLeft(n) == number.rotateLeft(n % 8)`\n
```

```
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class,  
ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\npublic inline fun UByte.rotateLeft(bitCount:  
Int): UByte = toByte().rotateLeft(bitCount).toUByte()\n\n/**\n * Rotates the binary representation of this [UByte]  
number right by the specified [bitCount] number of bits.\n * The least significant bits pushed out from the right side  
reenter the number as the most significant bits on the left side.\n * Rotating the number right by a negative bit  
count is the same as rotating it left by the negated bit count:\n * `number.rotateRight(-n) == number.rotateLeft(n)`\n *\n * Rotating by a multiple of [UByte.SIZE_BITS] (8) returns the same number, or more generally\n *
```

```

`number.rotateRight(n) == number.rotateRight(n % 8)`\n
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class,
ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\npublic inline fun UByte.rotateRight(bitCount:
Int): UByte = toByte().rotateRight(bitCount).toUByte()\n\n/**\n * Counts the number of set bits in the binary
representation of this [UShort] number.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UShort.countOneBits(): Int =
toUInt().countOneBits()\n\n/**\n * Counts the number of consecutive most significant bits that are zero in the
binary representation of this [UShort] number.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UShort.countLeadingZeroBits(): Int =
toShort().countLeadingZeroBits()\n\n/**\n * Counts the number of consecutive least significant bits that are zero
in the binary representation of this [UShort] number.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UShort.countTrailingZeroBits(): Int =
toShort().countTrailingZeroBits()\n\n/**\n * Returns a number having a single bit set in the position of the most
significant set bit of this [UShort] number,\n * or zero, if this number is zero.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UShort.takeHighestOneBit(): UShort =
toInt().takeHighestOneBit().toUShort()\n\n/**\n * Returns a number having a single bit set in the position of the
least significant set bit of this [UShort] number,\n * or zero, if this number is zero.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class,
ExperimentalStdlibApi::class)\n@kotlin.internal.InlineOnly\npublic inline fun UShort.takeLowestOneBit(): UShort =
toInt().takeLowestOneBit().toUShort()\n\n/**\n * Rotates the binary representation of this [UShort] number left
by the specified [bitCount] number of bits.\n * The most significant bits pushed out from the left side reenter the
number as the least significant bits on the right side.\n * Rotating the number left by a negative bit count is the
same as rotating it right by the negated bit count:\n * `number.rotateLeft(-n) == number.rotateRight(n)`\n * Rotating by a multiple of [UShort.SIZE_BITS] (16) returns the same number, or more generally\n *
`number.rotateLeft(n) == number.rotateLeft(n % 16)`\n
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class,
ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\npublic inline fun UShort.rotateLeft(bitCount:
Int): UShort = toShort().rotateLeft(bitCount).toUShort()\n\n/**\n * Rotates the binary representation of this
[UShort] number right by the specified [bitCount] number of bits.\n * The least significant bits pushed out from the
right side reenter the number as the most significant bits on the left side.\n * Rotating the number right by a
negative bit count is the same as rotating it left by the negated bit count:\n * `number.rotateRight(-n) ==
number.rotateLeft(n)`\n * Rotating by a multiple of [UShort.SIZE_BITS] (16) returns the same number, or more
generally\n * `number.rotateRight(n) == number.rotateRight(n % 16)`\n
*\n@SinceKotlin("1.6")\n@WasExperimental(ExperimentalStdlibApi::class,
ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly\npublic inline fun UShort.rotateRight(bitCount:
Int): UShort = toShort().rotateRight(bitCount).toUShort()\n\n"/*\n * Copyright 2010-2021 JetBrains s.r.o. and Kotlin
Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be
found in the license/LICENSE.txt file.\n *\n@kotlin.internal\npackage kotlin.internal\n// (a - b) mod c\nprivate fun
differenceModulo(a: UInt, b: UInt, c: UInt): UInt {\n    val ac = a % c\n    val bc = b % c\n    return if (ac >= bc) ac -
bc else ac - bc + c\n}\n\nprivate fun differenceModulo(a: ULong, b: ULong, c: ULong): ULong {\n    val ac = a %
c\n    val bc = b % c\n    return if (ac >= bc) ac - bc else ac - bc + c\n}\n\n/**\n * Calculates the final element of a
bounded arithmetic progression, i.e. the last element of the progression which is in the range\n * from [start] to [end]
in case of a positive [step], or from [end] to [start] in case of a negative\n * [step].\n * No validation on passed
parameters is performed. The given parameters should satisfy the condition:\n * - either `step > 0` and `start <=

```

```

end` ,\n * - or `step < 0` and `start >= end`.\n *\n * @param start first element of the progression\n * @param end
ending bound for the progression\n * @param step increment, or difference of successive elements in the
progression\n * @return the final element of the progression\n * @suppress\n
*\n@PublishedApi\n@SinceKotlin("1.3")\ninternal fun getProgressionLastElement(start: UInt, end: UInt, step:
Int): UInt = when {\n    step > 0 -> if (start >= end) end else end - differenceModulo(end, start, step.toUInt())\n
step < 0 -> if (start <= end) end else end + differenceModulo(start, end, (-step).toUInt())\n    else -> throw
kotlin.IllegalArgumentException("Step is zero.")\n}\n\n/**\n * Calculates the final element of a bounded
arithmetic progression, i.e. the last element of the progression which is in the range\n * from [start] to [end] in case
of a positive [step], or from [end] to [start] in case of a negative\n * [step].\n *\n * No validation on passed
parameters is performed. The given parameters should satisfy the condition:\n *\n * - either `step > 0` and `start <=
end` ,\n * - or `step < 0` and `start >= end`.\n *\n * @param start first element of the progression\n * @param end
ending bound for the progression\n * @param step increment, or difference of successive elements in the
progression\n * @return the final element of the progression\n * @suppress\n
*\n@PublishedApi\n@SinceKotlin("1.3")\ninternal fun getProgressionLastElement(start: ULong, end: ULong,
step: Long): ULong = when {\n    step > 0 -> if (start >= end) end else end - differenceModulo(end, start,
step.toULong())\n    step < 0 -> if (start <= end) end else end + differenceModulo(start, end, (-step).toULong())\n
else -> throw kotlin.IllegalArgumentException("Step is zero.")\n}\n\n/*\n * Copyright 2010-2021 JetBrains s.r.o.
and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license
that can be found in the license/LICENSE.txt file.\n *\n@file:kotlin.jvm.JvmName("UStringsKt") // string
representation of unsigned numbers\n\npackage kotlin.text\n\n/**\n * Returns a string representation of this [Byte]
value in the specified [radix].\n *\n * @throws IllegalArgumentException when [radix] is not a valid radix for
number to string conversion.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly
\npublic /*inline*/ fun UByte.toString(radix: Int): String = this.toInt().toString(radix)\n\n/**\n * Returns a string
representation of this [Short] value in the specified [radix].\n *\n * @throws IllegalArgumentException when [radix]
is not a valid radix for number to string conversion.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly
\npublic /*inline*/ fun UShort.toString(radix: Int): String = this.toInt().toString(radix)\n\n/**\n * Returns a string
representation of this [Int] value in the specified [radix].\n *\n * @throws IllegalArgumentException when [radix] is
not a valid radix for number to string conversion.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\n@kotlin.internal.InlineOnly
\npublic /*inline*/ fun UInt.toString(radix: Int): String = this.toLong().toString(radix)\n\n/**\n * Returns a string
representation of this [Long] value in the specified [radix].\n *\n * @throws IllegalArgumentException when [radix]
is not a valid radix for number to string conversion.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
ULong.toString(radix: Int): String = ulongToString(this.toLong(), checkRadix(radix))\n\n/**\n * Parses the string
as a signed [UByte] number and returns the result.\n *\n * @throws NumberFormatException if the string is not a valid
representation of a number.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun String.toUByte():
UByte = toUByteOrNull() ?: numberFormatException(this)\n\n/**\n * Parses the string as a signed [UByte] number and
returns the result.\n *\n * @throws NumberFormatException if the string is not a valid representation of a number.\n *
@throws IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
String.toUByte(radix: Int): UByte = toUByteOrNull(radix) ?: numberFormatException(this)\n\n/**\n * Parses the
string as a [UShort] number and returns the result.\n *\n * @throws NumberFormatException if the string is not a valid
representation of a number.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun String.toUShort():
UShort = toUShortOrNull() ?: numberFormatException(this)\n\n/**\n * Parses the string as a [UShort] number and

```

```

returns the result.\n * @throws NumberFormatException if the string is not a valid representation of a number.\n *
@throws IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
String.toUShort(radix: Int): UShort = toUShortOrNull(radix) ?: numberFormatError(this)\n\n/**\n * Parses the
string as an [UInt] number and returns the result.\n * @throws NumberFormatException if the string is not a valid
representation of a number.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun String.toUInt():
UInt = toUIntOrNull() ?: numberFormatError(this)\n\n/**\n * Parses the string as an [UInt] number and returns the
result.\n * @throws NumberFormatException if the string is not a valid representation of a number.\n * @throws
IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
String.toUInt(radix: Int): UInt = toUIntOrNull(radix) ?: numberFormatError(this)\n\n/**\n * Parses the string as a
[ULong] number and returns the result.\n * @throws NumberFormatException if the string is not a valid
representation of a number.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun String.toULong():
ULong = toULongOrNull() ?: numberFormatError(this)\n\n/**\n * Parses the string as a [ULong] number and
returns the result.\n * @throws NumberFormatException if the string is not a valid representation of a number.\n *
@throws IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
String.toULong(radix: Int): ULong = toULongOrNull(radix) ?: numberFormatError(this)\n\n\n\n\n\n\n/**\n * Parses
the string as an [UByte] number and returns the result\n * or `null` if the string is not a valid representation of a
number.\n */\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
String.toUByteOrNull(): UByte? = toUByteOrNull(radix = 10)\n\n/**\n * Parses the string as an [UByte] number
and returns the result\n * or `null` if the string is not a valid representation of a number.\n */\n * @throws
IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
String.toUByteOrNull(radix: Int): UByte? {\n    val int = this.toUIntOrNull(radix) ?: return null\n    if (int >
UByte.MAX_VALUE) return null\n    return int.toUByte()\n}\n\n/**\n * Parses the string as an [UShort] number
and returns the result\n * or `null` if the string is not a valid representation of a number.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
String.toUShortOrNull(): UShort? = toUShortOrNull(radix = 10)\n\n/**\n * Parses the string as an [UShort] number
and returns the result\n * or `null` if the string is not a valid representation of a number.\n */\n * @throws
IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
String.toUShortOrNull(radix: Int): UShort? {\n    val int = this.toUIntOrNull(radix) ?: return null\n    if (int >
UShort.MAX_VALUE) return null\n    return int.toUShort()\n}\n\n/**\n * Parses the string as an [UInt] number and
returns the result\n * or `null` if the string is not a valid representation of a number.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
String.toUIntOrNull(): UInt? = toUIntOrNull(radix = 10)\n\n/**\n * Parses the string as an [UInt] number and
returns the result\n * or `null` if the string is not a valid representation of a number.\n */\n * @throws
IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n
*\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun
String.toUIntOrNull(radix: Int): UInt? {\n    checkRadix(radix)\n\n    val length = this.length\n    if (length == 0)
return null\n\n    val limit: UInt = UInt.MAX_VALUE\n    val start: Int\n\n    val firstChar = this[0]\n    if (firstChar
< '0') {\n        if (length == 1 || firstChar != '+') return null\n        start = 1\n    } else {\n        start = 0\n    }\n\n    val
limitForMaxRadix = 119304647u // limit / 36\n\n    var limitBeforeMul = limitForMaxRadix\n    val uradix =
radix.toUInt()\n    var result = 0u\n    for (i in start until length) {\n        val digit = digitOf(this[i], radix)\n\n        if
(digit < 0) return null\n        if (result > limitBeforeMul) {\n            if (limitBeforeMul == limitForMaxRadix) {\n

```

```

        limitBeforeMul = limit / uradix\n\n        if (result > limitBeforeMul) {\n            return null\n        }\n    } else {\n        return null\n    }\n}\n\n    result *= uradix\n\n    val beforeAdding = result\n    result += digit.toUInt()\n    if (result < beforeAdding) return null // overflow has happened\n}\n\nreturn result\n}\n\n/**\n * Parses the string as an [ULong] number and returns the result\n * or `null` if the string is not a valid representation of a number.\n */\n\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun\nString.toULongOrNull(): ULong? = toULongOrNull(radix = 10)\n\n/**\n * Parses the string as an [ULong] number and returns the result\n * or `null` if the string is not a valid representation of a number.\n * @throws\n * IllegalArgumentException when [radix] is not a valid radix for string to number conversion.\n */\n\n@SinceKotlin("1.5")\n@WasExperimental(ExperimentalUnsignedTypes::class)\npublic fun\nString.toULongOrNull(radix: Int): ULong? {\n    checkRadix(radix)\n\n    val length = this.length\n    if (length == 0) return null\n\n    val limit: ULong = ULong.MAX_VALUE\n    val start: Int\n    val firstChar = this[0]\n    if (firstChar < '0') {\n        if (length == 1 || firstChar != '+') return null\n        start = 1\n    } else {\n        start = 0\n    }\n\n    val limitForMaxRadix = 512409557603043100uL // limit / 36\n    var limitBeforeMul = limitForMaxRadix\n    val uradix = radix.toULong()\n    var result = 0uL\n    for (i in start until length) {\n        val digit = digitOf(this[i], radix)\n        if (digit < 0) return null\n        if (result > limitBeforeMul) {\n            if (limitBeforeMul == limitForMaxRadix) {\n                limitBeforeMul = limit / uradix\n\n                if (result > limitBeforeMul) {\n                    return null\n                }\n            } else {\n                return null\n            }\n        }\n        result *= uradix\n        val beforeAdding = result\n        result += digit.toUInt()\n        if (result < beforeAdding) return null // overflow has happened\n    }\n\n    return result\n}\n\n"/**\n * Copyright 2010-2018\n * JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the\n * Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n@file:Suppress("INVISIBLE_REFERENCE", "INVISIBLE_MEMBER")\npackage kotlin\n\nimport\nkotlin.annotation.AnnotationTarget\n\nimport kotlin.internal.RequireKotlin\n\nimport\nkotlin.internal.RequireKotlinVersionKind\n\n/**\n * Marks the API that is dependent on the experimental unsigned\n * types, including those types themselves.\n * Usages of such API will be reported as warnings unless an explicit\n * opt-in with\n * the [OptIn] annotation, e.g. `@OptIn(ExperimentalUnsignedTypes::class)`\n * or with the `-Xopt-in=kotlin.ExperimentalUnsignedTypes`\n * compiler option is given.\n * It's recommended to propagate the\n * experimental status to the API that depends on unsigned types by\n * annotating it with this annotation.\n */\n\n@Suppress("DEPRECATION")\n@Experimental(level =\n    Experimental.Level.WARNING)\n@RequiresOptIn(level =\n    RequiresOptIn.Level.WARNING)\n@MustBeDocumented\n@Target(CLASS, ANNOTATION_CLASS,\n    PROPERTY, FIELD, LOCAL_VARIABLE, VALUE_PARAMETER, CONSTRUCTOR,\n    FUNCTION, PROPERTY_GETTER, PROPERTY_SETTER,\n    TYPEALIAS)\n@Retention(AnnotationRetention.BINARY)\n@RequireKotlin("1.2.50", versionKind =\n    RequireKotlinVersionKind.COMPILER_VERSION)\npublic annotation class ExperimentalUnsignedTypes\n\n"/**\n * Copyright 2010-2018 JetBrains s.r.o. and Kotlin Programming Language contributors.\n * Use of this source code is governed by the Apache 2.0 license that can be found in the license/LICENSE.txt file.\n */\n\n@file:kotlin.jvm.JvmMultifileClass\n@file:kotlin.jvm.JvmName("MathKt")\npackage\nkotlin.math\n\n// constants, can't use them from nativeMath as they are not constants there\n\n/**\n * Ratio of the circumference of a circle to its diameter, approximately 3.14159.\n */\n@SinceKotlin("1.2")\npublic const val PI: Double = 3.141592653589793\n\n/**\n * Base of the natural logarithms, approximately 2.71828.\n */\n@SinceKotlin("1.2")\npublic const val E: Double = 2.718281828459045\n\n// region =====\n\nDouble Math =====\n\n/**\n * Computes the sine of the angle [x]\n * given in radians.\n * Special cases:\n * - `sin(NaN|+Inf|-Inf)` is `NaN`\n */\n@SinceKotlin("1.2")\npublic expect fun sin(x: Double): Double\n\n/**\n * Computes the cosine of the angle [x]\n * given in radians.\n * Special cases:\n * - `cos(NaN|+Inf|-Inf)` is `NaN`\n */\n@SinceKotlin("1.2")\npublic expect fun cos(x: Double): Double\n\n/**\n * Computes the tangent of the angle [x]\n * given in radians.\n * Special cases:\n * - `tan(NaN|+Inf|-`

```

`Inf)` is `NaN`.  
`@SinceKotlin("1.2")` public expect fun `tan(x: Double): Double`.  
 Computes the arc sine of the value `[x]`; the returned value is an angle in the range from `-PI/2` to `PI/2` radians.  
 Special cases:  
 - `asin(x)` is `NaN`, when `abs(x) > 1` or `x` is `NaN`.  
`@SinceKotlin("1.2")` public expect fun `asin(x: Double): Double`.  
 Computes the arc cosine of the value `[x]`; the returned value is an angle in the range from `0.0` to `PI` radians.  
 Special cases:  
 - `acos(x)` is `NaN`, when `abs(x) > 1` or `x` is `NaN`.  
`@SinceKotlin("1.2")` public expect fun `acos(x: Double): Double`.  
 Computes the arc tangent of the value `[x]`; the returned value is an angle in the range from `-PI/2` to `PI/2` radians.  
 Special cases:  
 - `atan(NaN)` is `NaN`.  
`@SinceKotlin("1.2")` public expect fun `atan(x: Double): Double`.  
 Returns the angle `theta` of the polar coordinates `(r, theta)` that correspond to the rectangular coordinates `(x, y)` by computing the arc tangent of the value `[y] / [x]`; the returned value is an angle in the range from `-PI` to `PI` radians.  
 Special cases:  
 - `atan2(0.0, 0.0)` is `0.0`.  
 - `atan2(0.0, x)` is `0.0` for `x > 0` and `PI` for `x < 0`.  
 - `atan2(-0.0, x)` is `-0.0` for `x > 0` and `-PI` for `x < 0`.  
 - `atan2(y, +Inf)` is `0.0` for `0 < y < +Inf` and `-0.0` for `-Inf < y < 0`.  
 - `atan2(y, -Inf)` is `PI` for `0 < y < +Inf` and `-PI` for `-Inf < y < 0`.  
 - `atan2(y, 0.0)` is `PI/2` for `y > 0` and `-PI/2` for `y < 0`.  
 - `atan2(+Inf, x)` is `PI/2` for finite `x`.  
 - `atan2(-Inf, x)` is `-PI/2` for finite `x`.  
 - `atan2(NaN, x)` and `atan2(y, NaN)` is `NaN`.  
`@SinceKotlin("1.2")` public expect fun `atan2(y: Double, x: Double): Double`.  
 Computes the hyperbolic sine of the value `[x]`.  
 Special cases:  
 - `sinh(NaN)` is `NaN`.  
 - `sinh(+Inf)` is `+Inf`.  
 - `sinh(-Inf)` is `-Inf`.  
`@SinceKotlin("1.2")` public expect fun `sinh(x: Double): Double`.  
 Computes the hyperbolic cosine of the value `[x]`.  
 Special cases:  
 - `cosh(NaN)` is `NaN`.  
 - `cosh(+Inf)` is `+Inf`.  
`@SinceKotlin("1.2")` public expect fun `cosh(x: Double): Double`.  
 Computes the hyperbolic tangent of the value `[x]`.  
 Special cases:  
 - `tanh(NaN)` is `NaN`.  
 - `tanh(+Inf)` is `1.0`.  
 - `tanh(-Inf)` is `-1.0`.  
`@SinceKotlin("1.2")` public expect fun `tanh(x: Double): Double`.  
 Computes the inverse hyperbolic sine of the value `[x]`.  
 The returned value is `y` such that `sinh(y) == x`.  
 Special cases:  
 - `asinh(NaN)` is `NaN`.  
 - `asinh(+Inf)` is `+Inf`.  
 - `asinh(-Inf)` is `-Inf`.  
`@SinceKotlin("1.2")` public expect fun `asinh(x: Double): Double`.  
 Computes the inverse hyperbolic cosine of the value `[x]`.  
 The returned value is positive `y` such that `cosh(y) == x`.  
 Special cases:  
 - `acosh(NaN)` is `NaN`.  
 - `acosh(x)` is `NaN` when `x < 1`.  
 - `acosh(+Inf)` is `+Inf`.  
`@SinceKotlin("1.2")` public expect fun `acosh(x: Double): Double`.  
 Computes the inverse hyperbolic tangent of the value `[x]`.  
 The returned value is `y` such that `tanh(y) == x`.  
 Special cases:  
 - `atanh(NaN)` is `NaN`.  
 - `atanh(x)` is `NaN` when `x > 1` or `x < -1`.  
 - `atanh(1.0)` is `+Inf`.  
 - `atanh(-1.0)` is `-Inf`.  
`@SinceKotlin("1.2")` public expect fun `atanh(x: Double): Double`.  
 Computes `sqrt(x^2 + y^2)` without intermediate overflow or underflow.  
 Special cases:  
 - returns `+Inf` if any of arguments is infinite.  
 - returns `NaN` if any of arguments is `NaN` and the other is not infinite.  
`@SinceKotlin("1.2")` public expect fun `hypot(x: Double, y: Double): Double`.  
 Computes the positive square root of the value `[x]`.  
 Special cases:  
 - `sqrt(x)` is `NaN` when `x < 0` or `x` is `NaN`.  
`@SinceKotlin("1.2")` public expect fun `sqrt(x: Double): Double`.  
 Computes Euler's number `e` raised to the power of the value `[x]`.  
 Special cases:  
 - `exp(NaN)` is `NaN`.  
 - `exp(+Inf)` is `+Inf`.  
 - `exp(-Inf)` is `0.0`.  
`@SinceKotlin("1.2")` public expect fun `exp(x: Double): Double`.  
 Computes `exp(x) - 1`.  
 This function can be implemented to produce more precise result for `[x]` near zero.  
 Special cases:  
 - `expm1(NaN)` is `NaN`.  
 - `expm1(+Inf)` is `+Inf`.  
 - `expm1(-Inf)` is `-1.0`.  
 @see `[exp]` function.  
`@SinceKotlin("1.2")` public expect fun `expm1(x: Double): Double`.  
 Computes the logarithm of the value `[x]` to the given `[base]`.  
 Special cases:  
 - `log(x, b)` is `NaN` if either `x` or `b` are `NaN`.  
 - `log(x, b)` is `NaN` when `x < 0` or `b <= 0` or `b == 1.0`.  
 - `log(+Inf, +Inf)` is `NaN`.  
 - `log(+Inf, b)` is `+Inf` for `b > 1` and `-Inf` for `b < 1`.  
 - `log(0.0, b)` is `-Inf` for `b > 1` and `+Inf` for `b < 1`.  
 See also logarithm functions for common fixed bases: `[ln]`, `[log10]` and `[log2]`.  
`@SinceKotlin("1.2")` public expect fun `log(x: Double, base: Double): Double`.  
 Computes the natural logarithm (base `E`) of the value `[x]`.  
 Special cases:  
 - `ln(NaN)` is `NaN`.  
 - `ln(x)` is `NaN` when `x < 0.0`.  
 - `ln(+Inf)` is `+Inf`.  
 - `ln(0.0)` is `-Inf`.  
`@SinceKotlin("1.2")` public expect fun

`ln(x: Double): Double` Computes the common logarithm (base 10) of the value [x].  
 @see [ln] function for special cases.  
`public expect fun log10(x: Double): Double` Computes the binary logarithm (base 2) of the value [x].  
 @see [ln] function for special cases.  
`public expect fun log2(x: Double): Double` Computes  $\ln(x + 1)$ .  
 This function can be implemented to produce more precise result for [x] near zero.  
 Special cases:  
 $\ln(\text{NaN})$  is `NaN`  
 $\ln(x)$  is `NaN` where  $x < -1.0$   
 $\ln(-1.0)$  is `-Inf`  
 $\ln(+Inf)$  is `+Inf`  
 @see [ln] function  
 @see [expm1] function  
`public expect fun ln1p(x: Double): Double` Rounds the given value [x] to an integer towards positive infinity.  
 @return the smallest double value that is greater than or equal to the given value [x] and is a mathematical integer.  
 Special cases:  
 $\text{ceil}(x)$  is `x` where `x` is `NaN` or `+Inf` or `-Inf` or already a mathematical integer.  
`public expect fun ceil(x: Double): Double` Rounds the given value [x] to an integer towards negative infinity.  
 @return the largest double value that is smaller than or equal to the given value [x] and is a mathematical integer.  
 Special cases:  
 $\text{floor}(x)$  is `x` where `x` is `NaN` or `+Inf` or `-Inf` or already a mathematical integer.  
`public expect fun floor(x: Double): Double` Rounds the given value [x] to an integer towards zero.  
 @return the value [x] having its fractional part truncated.  
 Special cases:  
 $\text{truncate}(x)$  is `x` where `x` is `NaN` or `+Inf` or `-Inf` or already a mathematical integer.  
`public expect fun truncate(x: Double): Double` Rounds the given value [x] towards the closest integer with ties rounded towards even integer.  
 Special cases:  
 $\text{round}(x)$  is `x` where `x` is `NaN` or `+Inf` or `-Inf` or already a mathematical integer.  
`public expect fun round(x: Double): Double` Returns the absolute value of the given value [x].  
 Special cases:  
 $\text{abs}(\text{NaN})$  is `NaN`  
 @see `absoluteValue` extension property for [Double]  
`public expect fun abs(x: Double): Double` Returns the sign of the given value [x]:  
`-1.0` if the value is negative,  
`0.0` if the value is zero,  
`1.0` if the value is positive  
 Special case:  
 $\text{sign}(\text{NaN})$  is `NaN`  
`public expect fun sign(x: Double): Double` Returns the smaller of two values.  
 If either value is `NaN`, then the result is `NaN`.  
`public expect fun min(a: Double, b: Double): Double` Returns the greater of two values.  
 If either value is `NaN`, then the result is `NaN`.  
`public expect fun max(a: Double, b: Double): Double` Raises this value to the power [x].  
 Special cases:  
 $b.\text{pow}(0.0)$  is `1.0`  
 $b.\text{pow}(1.0) == b$   
 $b.\text{pow}(\text{NaN})$  is `NaN`  
 $\text{NaN}.\text{pow}(x)$  is `NaN` for  $x \neq 0.0$   
 $b.\text{pow}(\text{Inf})$  is `NaN` for  $\text{abs}(b) == 1.0$   
 $b.\text{pow}(x)$  is `NaN` for  $b < 0$  and `x` is finite and not an integer  
`public expect fun Double.pow(x: Double): Double` Raises this value to the integer power [n].  
 See the other overload of [pow] for details.  
`public expect fun Double.pow(n: Int): Double` Returns the absolute value of this value.  
 Special cases:  
 $\text{NaN}.\text{absoluteValue}$  is `NaN`  
 @see `abs` function  
`public expect val Double.absoluteValue: Double` Returns the sign of this value:  
`-1.0` if the value is negative,  
`0.0` if the value is zero,  
`1.0` if the value is positive  
 Special case:  
 $\text{NaN}.\text{sign}$  is `NaN`  
`public expect val Double.sign: Double` Returns this value with the sign bit same as of the [sign] value.  
 If [sign] is `NaN` the sign of the result is undefined.  
`public expect fun Double.withSign(sign: Double): Double` Returns this value with the sign bit same as of the [sign] value.  
`public expect fun Double.withSign(sign: Int): Double` Returns the ulp (unit in the last place) of this value.  
 An ulp is a positive distance between this value and the next nearest [Double] value larger in magnitude.  
 Special Cases:  
 $\text{NaN}.\text{ulp}$  is `NaN`  
 $x.\text{ulp}$  is `+Inf` when `x` is `+Inf` or `-Inf`  
 $0.0.\text{ulp}$  is `Double.MIN_VALUE`  
`public expect val Double.ulp: Double` Returns the [Double] value nearest to this value in direction of positive infinity.  
`public expect fun Double.nextUp(): Double` Returns the [Double] value nearest to this value in direction of negative infinity.  
`public expect fun Double.nextDown(): Double` Returns the [Double] value

nearest to this value in direction from this value towards the value [to].  
Special cases:  
`x.nextTowards(y)` is `NaN` if either `x` or `y` are `NaN`  
`x.nextTowards(x) == x`  
@SinceKotlin("1.2")\npublic expect fun Double.nextTowards(to: Double): Double  
Rounds this [Double] value to the nearest integer and converts the result to [Int].  
Ties are rounded towards positive infinity.  
Special cases:  
`x.roundToInt() == Int.MAX_VALUE` when `x > Int.MAX_VALUE`  
`x.roundToInt() == Int.MIN_VALUE` when `x < Int.MIN_VALUE`  
@throws IllegalArgumentException when this value is `NaN`  
@SinceKotlin("1.2")\npublic expect fun Double.roundToInt(): Int  
Rounds this [Double] value to the nearest integer and converts the result to [Long].  
Ties are rounded towards positive infinity.  
Special cases:  
`x.roundToLong() == Long.MAX_VALUE` when `x > Long.MAX_VALUE`  
`x.roundToLong() == Long.MIN_VALUE` when `x < Long.MIN_VALUE`  
@throws IllegalArgumentException when this value is `NaN`  
@SinceKotlin("1.2")\npublic expect fun Double.roundToLong(): Long  
// endregion  
// region ===== Float Math  
===== \n\n\*\* Computes the sine of the angle [x] given in radians.  
Special cases:  
`sin(NaN|+Inf|-Inf)` is `NaN`  
@SinceKotlin("1.2")\npublic expect fun sin(x: Float): Float  
Computes the cosine of the angle [x] given in radians.  
Special cases:  
`cos(NaN|+Inf|-Inf)` is `NaN`  
@SinceKotlin("1.2")\npublic expect fun cos(x: Float): Float  
Computes the tangent of the angle [x] given in radians.  
Special cases:  
`tan(NaN|+Inf|-Inf)` is `NaN`  
@SinceKotlin("1.2")\npublic expect fun tan(x: Float): Float  
Computes the arc sine of the value [x];  
the returned value is an angle in the range from  $-\pi/2$  to  $\pi/2$  radians.  
Special cases:  
`asin(x)` is `NaN`, when `abs(x) > 1` or x is `NaN`  
@SinceKotlin("1.2")\npublic expect fun asin(x: Float): Float  
Computes the arc cosine of the value [x];  
the returned value is an angle in the range from  $0.0$  to  $\pi$  radians.  
Special cases:  
`acos(x)` is `NaN`, when `abs(x) > 1` or x is `NaN`  
@SinceKotlin("1.2")\npublic expect fun acos(x: Float): Float  
Computes the arc tangent of the value [x];  
the returned value is an angle in the range from  $-\pi/2$  to  $\pi/2$  radians.  
Special cases:  
`atan(NaN)` is `NaN`  
@SinceKotlin("1.2")\npublic expect fun atan(x: Float): Float  
Returns the angle  $\theta$  of the polar coordinates  $(r, \theta)$  that correspond  
to the rectangular coordinates  $(x, y)$  by computing the arc tangent of the value  $[y] / [x]$ ;  
the returned value is an angle in the range from  $-\pi$  to  $\pi$  radians.  
Special cases:  
`atan2(0.0, 0.0)` is  $0.0$   
`atan2(0.0, x)` is  $0.0$  for  $x > 0$  and  $\pi$  for  $x < 0$   
`atan2(-0.0, x)` is  $-0.0$  for  $x > 0$  and  $-\pi$  for  $x < 0$   
`atan2(y, +Inf)` is  $0.0$  for  $0 < y < +Inf$  and  $-0.0$  for  $-Inf < y < 0$   
`atan2(y, -Inf)` is  $\pi$  for  $0 < y < +Inf$  and  $-\pi$  for  $-Inf < y < 0$   
`atan2(y, 0.0)` is  $\pi/2$  for  $y > 0$  and  $-\pi/2$  for  $y < 0$   
`atan2(+Inf, x)` is  $\pi/2$  for finite  $x$   
`atan2(-Inf, x)` is  $-\pi/2$  for finite  $x$   
`atan2(NaN, x)` and `atan2(y, NaN)` is `NaN`  
@SinceKotlin("1.2")\npublic expect fun atan2(y: Float, x: Float): Float  
Computes the hyperbolic sine of the value [x].  
Special cases:  
`sinh(NaN)` is `NaN`  
`sinh(+Inf)` is  $+Inf$   
`sinh(-Inf)` is  $-Inf$   
@SinceKotlin("1.2")\npublic expect fun sinh(x: Float): Float  
Computes the hyperbolic cosine of the value [x].  
Special cases:  
`cosh(NaN)` is `NaN`  
`cosh(+Inf|-Inf)` is  $+Inf$   
@SinceKotlin("1.2")\npublic expect fun cosh(x: Float): Float  
Computes the hyperbolic tangent of the value [x].  
Special cases:  
`tanh(NaN)` is `NaN`  
`tanh(+Inf)` is  $1.0$   
`tanh(-Inf)` is  $-1.0$   
@SinceKotlin("1.2")\npublic expect fun tanh(x: Float): Float  
Computes the inverse hyperbolic sine of the value [x].  
The returned value is  $y$  such that  $\sinh(y) == x$ .  
Special cases:  
`asinh(NaN)` is `NaN`  
`asinh(+Inf)` is  $+Inf$   
`asinh(-Inf)` is  $-Inf$   
@SinceKotlin("1.2")\npublic expect fun asinh(x: Float): Float  
Computes the inverse hyperbolic cosine of the value [x].  
The returned value is positive  $y$  such that  $\cosh(y) == x$ .  
Special cases:  
`acosh(NaN)` is `NaN`  
`acosh(x)` is `NaN` when  $x < 1$   
`acosh(+Inf)` is  $+Inf$   
@SinceKotlin("1.2")\npublic expect fun acosh(x: Float): Float  
Computes the inverse hyperbolic tangent of the value [x].  
The returned value is  $y$  such that  $\tanh(y) == x$ .  
Special cases:  
`tanh(NaN)` is `NaN`  
`tanh(x)` is `NaN` when  $x > 1$  or  $x < -1$   
`tanh(1.0)` is  $+Inf$   
`tanh(-1.0)` is  $-Inf$   
@SinceKotlin("1.2")\npublic expect fun atanh(x: Float): Float  
Computes  $\sqrt{x^2 +$

$y^2$ ) without intermediate overflow or underflow.  
`* Special cases:`  
`* - returns +Inf if any of arguments is infinite`  
`* - returns NaN if any of arguments is NaN and the other is not infinite`

`*\n@SinceKotlin("1.2")\npublic expect fun hypot(x: Float, y: Float): Float\n\n/**\n * Computes the positive square root of the value [x].\n * Special cases:\n * - sqrt(x) is NaN when x < 0 or x is NaN\n`

`*\n@SinceKotlin("1.2")\npublic expect fun sqrt(x: Float): Float\n\n/**\n * Computes Euler's number e raised to the power of the value [x].\n * Special cases:\n * - exp(NaN) is NaN\n * - exp(+Inf) is +Inf\n * - exp(-Inf) is 0.0\n`

`*\n@SinceKotlin("1.2")\npublic expect fun exp(x: Float): Float\n\n/**\n * Computes exp(x) - 1.  
* This function can be implemented to produce more precise result for [x] near zero.  
* Special cases:\n * - expm1(NaN) is NaN\n * - expm1(+Inf) is +Inf\n * - expm1(-Inf) is -1.0\n * @see [exp] function.\n`

`*\n@SinceKotlin("1.2")\npublic expect fun expm1(x: Float): Float\n\n/**\n * Computes the logarithm of the value [x] to the given [base].\n * Special cases:\n * - log(x, b) is NaN if either x or b are NaN\n * - log(x, b) is NaN when x < 0 or b <= 0 or b == 1.0\n * - log(+Inf, +Inf) is NaN\n * - log(+Inf, b) is +Inf for b > 1 and -Inf for b < 1\n * - log(0.0, b) is -Inf for b > 1 and +Inf for b > 1\n`

`* See also logarithm functions for common fixed bases: [ln], [log10] and [log2].\n`

`*\n@SinceKotlin("1.2")\npublic expect fun log(x: Float, base: Float): Float\n\n/**\n * Computes the natural logarithm (base E) of the value [x].\n * Special cases:\n * - ln(NaN) is NaN\n * - ln(x) is NaN when x < 0.0\n * - ln(+Inf) is +Inf\n * - ln(0.0) is -Inf\n`

`*\n@SinceKotlin("1.2")\npublic expect fun ln(x: Float): Float\n\n/**\n * Computes the common logarithm (base 10) of the value [x].\n * @see [ln] function for special cases.\n`

`*\n@SinceKotlin("1.2")\npublic expect fun log10(x: Float): Float\n\n/**\n * Computes the binary logarithm (base 2) of the value [x].\n * @see [ln] function for special cases.\n`

`*\n@SinceKotlin("1.2")\npublic expect fun log2(x: Float): Float\n\n/**\n * Computes ln(a + 1).  
* This function can be implemented to produce more precise result for [x] near zero.  
* Special cases:\n * - ln1p(NaN) is NaN\n * - ln1p(x) is NaN where x < -1.0\n * - ln1p(-1.0) is -Inf\n * - ln1p(+Inf) is +Inf\n * @see [ln] function\n * @see [expm1] function\n`

`*\n@SinceKotlin("1.2")\npublic expect fun ln1p(x: Float): Float\n\n/**\n * Rounds the given value [x] to an integer towards positive infinity.  
* @return the smallest Float value that is greater than or equal to the given value [x] and is a mathematical integer.  
* Special cases:\n * - ceil(x) is x where x is NaN or +Inf or -Inf or already a mathematical integer.\n`

`*\n@SinceKotlin("1.2")\npublic expect fun ceil(x: Float): Float\n\n/**\n * Rounds the given value [x] to an integer towards negative infinity.  
* @return the largest Float value that is smaller than or equal to the given value [x] and is a mathematical integer.  
* Special cases:\n * - floor(x) is x where x is NaN or +Inf or -Inf or already a mathematical integer.\n`

`*\n@SinceKotlin("1.2")\npublic expect fun floor(x: Float): Float\n\n/**\n * Rounds the given value [x] to an integer towards zero.  
* @return the value [x] having its fractional part truncated.  
* Special cases:\n * - truncate(x) is x where x is NaN or +Inf or -Inf or already a mathematical integer.\n`

`*\n@SinceKotlin("1.2")\npublic expect fun truncate(x: Float): Float\n\n/**\n * Rounds the given value [x] towards the closest integer with ties rounded towards even integer.  
* Special cases:\n * - round(x) is x where x is NaN or +Inf or -Inf or already a mathematical integer.\n`

`*\n@SinceKotlin("1.2")\npublic expect fun round(x: Float): Float\n\n/**\n * Returns the absolute value of the given value [x].  
* Special cases:\n * - abs(NaN) is NaN\n * @see absoluteValue extension property for [Float]\n`

`*\n@SinceKotlin("1.2")\npublic expect fun abs(x: Float): Float\n\n/**\n * Returns the sign of the given value [x]:  
* - -1.0 if the value is negative,\n * - zero if the value is zero,\n * - 1.0 if the value is positive\n`

`* Special case:\n * - sign(NaN) is NaN\n`

`*\n@SinceKotlin("1.2")\npublic expect fun sign(x: Float): Float\n\n/**\n * Returns the smaller of two values.  
* If either value is NaN, then the result is NaN.\n`

`*\n@SinceKotlin("1.2")\npublic expect fun min(a: Float, b: Float): Float\n\n/**\n * Returns the greater of two values.  
* If either value is NaN, then the result is NaN.\n`

`*\n@SinceKotlin("1.2")\npublic expect fun max(a: Float, b: Float): Float\n\n// extensions\n\n/**\n * Raises this value to the power [x].  
* Special cases:\n * - b.pow(0.0) is 1.0\n * - b.pow(1.0) == b\n * - b.pow(NaN) is NaN\n * - NaN.pow(x) is NaN for x != 0.0\n * - b.pow(Inf) is NaN for abs(b) == 1.0\n * - b.pow(x) is NaN for b < 0 and x is finite and not an integer\n`

`*\n@SinceKotlin("1.2")\npublic expect fun Float.pow(x: Float): Float\n\n/**\n * Raises`

this value to the integer power [n].  
 \* SinceKotlin("1.2")\npublic expect fun Float.pow(n: Int): Float\n \* Returns the absolute value of this value.  
 \* Special cases: - NaN.absoluteValue` is `NaN`  
 \* @see abs function  
 \* SinceKotlin("1.2")\npublic expect val Float.absoluteValue: Float\n \* Returns the sign of this value:  
 \* - -1.0` if the value is negative,  
 \* - zero if the value is zero,  
 \* - 1.0` if the value is positive  
 \* Special case: - NaN.sign` is `NaN`  
 \* SinceKotlin("1.2")\npublic expect val Float.sign: Float\n \* Returns this value with the sign bit same as of the [sign] value.  
 \* If [sign] is `NaN` the sign of the result is undefined.  
 \* SinceKotlin("1.2")\npublic expect fun Float.withSign(sign: Float): Float\n \* Returns this value with the sign bit same as of the [sign] value.  
 \* SinceKotlin("1.2")\npublic expect fun Float.withSign(sign: Int): Float\n \* Rounds this [Float] value to the nearest integer and converts the result to [Int].  
 \* Ties are rounded towards positive infinity.  
 \* Special cases: - x.roundToInt() == Int.MAX\_VALUE` when `x > Int.MAX\_VALUE`  
 \* - x.roundToInt() == Int.MIN\_VALUE` when `x < Int.MIN\_VALUE`  
 \* @throws IllegalArgumentException when this value is `NaN`  
 \* SinceKotlin("1.2")\npublic expect fun Float.roundToInt(): Int\n \* Rounds this [Float] value to the nearest integer and converts the result to [Long].  
 \* Ties are rounded towards positive infinity.  
 \* Special cases: - x.roundToLong() == Long.MAX\_VALUE` when `x > Long.MAX\_VALUE`  
 \* - x.roundToLong() == Long.MIN\_VALUE` when `x < Long.MIN\_VALUE`  
 \* @throws IllegalArgumentException when this value is `NaN`  
 \* SinceKotlin("1.2")\npublic expect fun Float.roundToLong(): Long\n // endregion  
 ===== Integer Math =====  
 \* Returns the absolute value of the given value [n].  
 \* Special cases: - `abs(Int.MIN\_VALUE)` is `Int.MIN\_VALUE` due to an overflow  
 \* @see absoluteValue extension property for [Int]  
 \* SinceKotlin("1.2")\npublic expect fun abs(n: Int): Int\n \* Returns the smaller of two values.  
 \* SinceKotlin("1.2")\npublic expect fun min(a: Int, b: Int): Int\n \* Returns the greater of two values.  
 \* SinceKotlin("1.2")\npublic expect fun max(a: Int, b: Int): Int\n \* Returns the absolute value of this value.  
 \* Special cases: - `Int.MIN\_VALUE.absoluteValue` is `Int.MIN\_VALUE` due to an overflow  
 \* @see abs function  
 \* SinceKotlin("1.2")\npublic expect val Int.absoluteValue: Int\n \* Returns the sign of this value:  
 \* - -1` if the value is negative,  
 \* - 0` if the value is zero,  
 \* - 1` if the value is positive  
 \* SinceKotlin("1.2")\npublic expect val Int.sign: Int\n \* Returns the absolute value of the given value [n].  
 \* Special cases: - `abs(Long.MIN\_VALUE)` is `Long.MIN\_VALUE` due to an overflow  
 \* @see absoluteValue extension property for [Long]  
 \* SinceKotlin("1.2")\npublic expect fun abs(n: Long): Long\n \* Returns the smaller of two values.  
 \* SinceKotlin("1.2")\npublic expect fun min(a: Long, b: Long): Long\n \* Returns the greater of two values.  
 \* SinceKotlin("1.2")\npublic expect fun max(a: Long, b: Long): Long\n \* Returns the absolute value of this value.  
 \* Special cases: - `Long.MIN\_VALUE.absoluteValue` is `Long.MIN\_VALUE` due to an overflow  
 \* @see abs function  
 \* SinceKotlin("1.2")\npublic expect val Long.absoluteValue: Long\n \* Returns the sign of this value:  
 \* - -1` if the value is negative,  
 \* - 0` if the value is zero,  
 \* - 1` if the value is positive  
 \* SinceKotlin("1.2")\npublic expect val Long.sign: Int\n //  
 endregion\n","names":[],"mappings":"AAWC,CAXA,yB;EACG,IAAI,OAAO,MAAO,KAAI,UAAW,IAAG,MAAM,IAA1C,C;IACI,MAAM,CAAC,QAAD,EA AW,CAAC,SAAD,CAAX,EA AW,B,OAAxB,C;SAEL,IAAI,OAAO,OAAQ,KAAI,QAAvB,C;IACD,OAAO,CAAC,MAAM,QAAP,C;;IAGP,IAAI,OAAQ,GAAE,E;IACd,OAAO,CAAC,IAAI,OAAL,C;;CAEd,CAAC,IAAD,EAAO,kB;EACJ,IAAI,IAAI,M;ECPZ,MAAM,eAAgB,GAAE,a;IACpB,OA AoD,CAA5C,KAAK,QAAQ,CAAC,CAAD,CAAI,IAAG,CAA E,YAAW,SAAW,KAA G,CAAC,OAAQ,KAAI,c;G ;EAGxE,MAAM,YAAa,GAAE,a;IACjB,OAAO,CAA E,YAAW,SAAU,IAAG,CAAC,OAAQ,KAAI,c;G;EAGID,M AAM,aAAc,GAAE,a;IACIB,OAAO,CAA E,YAAW,U;G;EAGxB,MAAM,YAAa,GAAE,a;IACjB,OAAO,CAA E,Y AAW,WAA Y,IAAG,CAAC,OAAQ,KAAI,W;G;EAGpD,MAAM,WAA Y,GAAE,a;IAC hB,OAAO,CAA E,YAAW, U;G;EAGxB,MAAM,aAAc,GAAE,a;IACIB,OAAO,CAA E,YAAW,Y;G;EAGxB,MAAM,cAAe,GAAE,a;IACnB, OAAO,CAA E,YAAW,Y;G;EAGxB,MAAM,YAAa,GAAE,a;IACjB,OAAO,KAAK,QAAQ,CAAC,CAAD,CAAI,IA AG,CAAC,OAAQ,KAAI,W;G;EAG5C,MAAM,QAAS,GAAE,a;IACb,OAAO,KAAK,QAAQ,CAAC,CAAD,CAA

I,IAAG,CAAC,CAAC,O;G;EAGjC,MAAM,WAAY,GAAE,a;IACbB,OAAO,KAAK,QAAQ,CAAC,CAAD,CAAI,IAAG,WAAW,OAAO,CAAC,CAAD,C;G;EAGjD,MAAM,cAAe,GAAE,a;IACnB,IAAI,CAAE,KAAI,IAAV,C;M AAgB,OAAO,M;IACvB,IAAI,WAAW,MAAM,YAAY,CAAC,CAAD,CAAI,GAAE,MAAM,aAAR,GAAwB,MA AM,S;IACnE,OAAO,GAAI,GAAE,KAAK,UAAU,IAAI,KAAK,CAAC,CAAD,EAAI,a;MAAc,OAAO,QAAQ,CA AC,CAAD,C;KAAjC,CAAwC,KAAK,CAAC,IAAD,CAAQ,GAAE,G;G;EAG/F,MAAM,kBAAmB,GAAE,e;IACv B,OAAO,MAAM,OAAO,YAAY,wBAAwB,CAAC,GAAD,C;G;EAG5D,MAAM,YAAa,GAAE,gB;IACjB,IAAI,C AAE,KAAI,CAAV,C;MACI,OAAO,I;KAEX,IAAI,CAAE,KAAI,IAAK,IAAG,CAAE,KAAI,IAAK,IAAG,CAAC, MAAM,WAAW,CAAC,CAAD,CAAI,IAAG,CAAC,OAAQ,KAAI,CAAC,OAAvE,C;MACI,OAAO,K;KAGX,KA AK,IAAI,IAAI,CAAR,EAAW,IAAI,CAAC,OAArB,EAA8B,CAAE,GAAE,CAAIC,EAAqC,CAAC,EAAtC,C;MA CI,IAAI,CAAC,MAAM,OAAO,CAAC,CAAC,CAAC,CAAD,CAAF,EAAO,CAAC,CAAC,CAAD,CAAR,CAAIB ,C;QACI,OAAO,K;IAGf,OAAO,I;G;EAGX,MAAM,gBAAiB,GAAE,gB;IACrB,OAAO,MAAM,OAAO,YAAY,s BAAsB,CAAC,CAAD,EAAI,CAAJ,C;G;EAG1D,MAAM,cAAe,GAAE,e;IACnB,IAAI,GAAI,KAAI,IAAZ,C;MA AkB,OAAO,C;IACzB,IAAI,SAAS,C;IACb,KAAK,IAAI,IAAI,CAAR,EAAW,IAAI,GAAG,OAAvB,EAAgC,CAA E,GAAE,CAApC,EAAuC,CAAC,EAAxC,C;MACI,MAAO,GAAqB,CAAjB,EAAG,GAAE,MAAO,GAAE,CAAG ,IAAE,MAAM,SAAS,CAAC,GAAG,CAAC,CAAD,CAAJ,CAAU,GAAE,C;IAE7D,OAAO,M;G;EAGX,MAAM, kBAAmB,GAAE,e;IACvB,OAAO,MAAM,OAAO,YAAY,wBAAwB,CAAC,GAAD,C;G;EAG5D,MAAM,mBAA oB,GAAE,iB;IACxB,KAAK,KAAK,CAAC,MAAM,gBAAP,C;G;ECpFd,MAAM,eAAgB,GAAE,mB;IACpB,CA AC,aAAc,GAAE,I;IACjB,OAAO,C;G;EAGX,MAAM,uBAAwB,GAAE,4C;IAC5B,MAAM,IAAK,GAAE,M;IAC b,MAAM,IAAK,GAAE,M;IACb,MAAM,aAAc,GAAE,I;IACtB,OAAO,mBAAmB,CAAC,MAAD,EAAS,MAAT, EAAiB,6BAA6B,CAAC,UAAD,CAA9C,C;G;EAG9B,iD;IACI,GAAG,WAAY,GAAE,sBAAsB,CAAC,OAAO,M AAO,KAAI,UAAW,GAAE,KAAK,QAAP,GAakB,KAAK,UAArD,C;IACvC,GAAG,YAAa,GAAE,G;IACIB,OA AO,G;G;EAGX,IAAI,gCAAgC,CACHC,UACa,QAAS,IAAT,wBAAqC,Y;IAC1C,OAAO,MAAM,OAAO,QAAQ,k B;GADvB,CADb,aAIe,QAAS,IAAT,wBAAqC,Y;IAC5C,OAAO,MAAM,OAAO,QAAQ,W;GADrB,CAJf,CADgC ,EAShC,UACa,QAAS,IAAT,wBAAqC,Y;IAC1C,OAAO,MAAM,OAAO,QAAQ,kB;GADvB,CADb,aAIe,QAAS,I AAT,wBAAqC,Y;IAC5C,OAAO,MAAM,OAAO,QAAQ,W;GADrB,CAJf,CATgC,C;EAmBpC,uC;IACI,IAAI,KA AK,MAAO,KAAI,IAApB,C;MACI,KAAK,MAAO,GAAE,aACE,CAAC,KAAK,qBAAqB,EAA3B,CADF,aAEC,I AFD,aAGC,EAHD,cAIE,EAJF,SAKH,EALG,iBAMK,EANL,C;KASIB,OAAO,KAAK,M;G;EChDhB,MAAM,QA AS,GAAE,a;IACb,OAAoB,CAAZ,CAAE,GAAE,KAAQ,KAAG,EAAG,IAAG,E;G;EAGjC,MAAM,OAAQ,GAA E,a;IACZ,OAAkB,CAAV,CAAE,GAAE,GAAM,KAAG,EAAG,IAAG,E;G;EAG/B,MAAM,OAAQ,GAAE,a;IAC Z,OAAO,CAAE,GAAE,K;G;EAGf,MAAM,aAAc,GAAE,a;IACIB,OAAO,CAAE,YAAW,MAAM,KAAM,GAAE, CAAF,GAAM,MAAM,KAAK,WAAW,CAAC,CAAD,C;G;EAGhE,MAAM,YAAa,GAAE,a;IACjB,OAAO,CAAE ,YAAW,MAAM,KAAM,GAAE,CAAC,MAAM,EAAT,GAAC,MAAM,YAAY,CAAC,CAAD,C;G;EAGpE,MAA M,cAAe,GAAE,a;IACnB,OAAO,MAAM,QAAQ,CAAC,MAAM,YAAY,CAAC,CAAD,CAAnB,C;G;EAGzB,MA AM,aAAc,GAAE,a;IACIB,OAAO,MAAM,OAAO,CAAC,MAAM,YAAY,CAAC,CAAD,CAAnB,C;G;EAGxB,M AAM,eAAgB,GAAE,a;IACpB,OAAO,CAAC,C;G;EAGZ,MAAM,aAAc,GAAE,a;IACIB,OAAO,MAAM,OAAO, CAAC,MAAM,YAAY,CAAC,CAAD,CAAnB,C;G;EAGxB,MAAM,YAAa,GAAE,a;IACjB,IAAI,CAAE,GAAE,U AAR,C;MAAoB,OAAO,U;IAC3B,IAAI,CAAE,GAAE,WAAR,C;MAAqB,OAAO,W;IAC5B,OAAO,CAAE,GAA E,C;G;EAGf,MAAM,YAAa,GAAE,a;IACjB,IAAI,CAAE,IAAG,IAAT,C;MAAe,OAAO,C;IACtB,IAAI,CAAE,Y AAW,MAAM,UAAvB,C;MAAmC,OAAO,C;IAC1C,OAAO,IAAI,MAAM,UAAV,CAAqB,CAArB,C;G;EAGX,M AAM,UAAW,GAAE,a;IACf,IAAI,CAAE,IAAG,IAAT,C;MAAe,OAAO,C;IACtB,OAAO,MAAM,OAAO,CAAC, CAAD,C;G;ECIDxB,MAAM,OAAQ,GAAE,sB;IACZ,IAAI,IAAK,IAAG,IAAZ,C;MACI,OAAO,IAAK,IAAG,I;K AGnB,IAAI,IAAK,IAAG,IAAZ,C;MACI,OAAO,K;KAGX,IAAI,IAAK,KAAI,IAAb,C;MACI,OAAO,IAAK,KAA LI;KAGpB,IAAI,OAAO,IAAK,KAAI,QAAS,IAAG,OAAO,IAAI,OAAQ,KAAI,UAAvD,C;MACI,OAAO,IAAI,O AAO,CAAC,IAAD,C;KAGtB,IAAI,OAAO,IAAK,KAAI,QAAS,IAAG,OAAO,IAAK,KAAI,QAahD,C;MACI,OA AO,IAAK,KAAI,IAAK,KAAI,IAAK,KAAI,CAAE,IAAG,CAAE,GAAE,IAAK,KAAI,CAAE,GAAE,IAAnC,C;K AGzB,OAAO,IAAK,KAAI,I;G;EAGpB,MAAM,SAAU,GAAE,e;IACd,IAAI,GAAI,IAAG,IAAX,C;MACI,OAAO, C;KAEX,IAAI,UAAU,OAAO,G;IACrB,IAAI,QAAS,KAAI,OAAjB,C;MACI,OAAO,UAAW,KAAI,OAAO,GAA G,SAAU,GAAE,GAAG,SAAS,EAAd,GAAMb,iBAAiB,CAAC,GAAD,C;KAEIF,IAAI,UAAW,KAAI,OAAnB,C;

MACI,OAAO,iBAAiB,CAAC,GAAD,C;KAE5B,IAAI,QAAS,KAAI,OAAjB,C;MACI,OAAO,MAAM,eAAe,CAA  
C,GAAD,C;KAEhC,IAAI,SAAU,KAAI,OAAiB,C;MACI,OAAO,MAAM,CAAC,GAAD,C;KAGjB,IAAI,MAAM,  
MAAM,CAAC,GAAD,C;IACHb,OAAO,iBAAiB,CAAC,GAAD,C;G;EAI5B,MAAM,SAAU,GAAE,a;IACd,IAAI,  
CAAE,IAAG,IAAT,C;MACI,OAAO,M;WAEN,IAAI,MAAM,WAAW,CAAC,CAAD,CAArB,C;MACD,OAAO,O  
;;MAGP,OAAO,CAAC,SAAS,E;;G;EAKzB,IAAI,WAAW,a;EAGf,IAAI,iCAAiC,sB;EAErC,gC;IACI,IAAI,EAAE  
,8BAA+B,IAAG,GAApC,CAAJ,C;MACI,IAAI,OAAQ,IAAI,OAAO,EAAG,GAAE,QAAU,GAAE,C;MACxC,MA  
AM,eAAe,CAAC,GAAD,EAAM,8BAAN,EAAsC,QAAU,IAAV,cAA4B,KAA5B,CAAtC,C;KAEzB,OAAO,GAA  
G,CAAC,8BAAD,C;G;EAGd,gC;IACI,IAAI,OAAO,C;IACX,KAAK,IAAI,IAAI,CAAb,EAAgB,CAAE,GAAE,G  
AAG,OAAvB,EAAgC,CAAC,EAAjC,C;MACI,IAAI,OAAQ,GAAG,WAAW,CAAC,CAAD,C;MAC1B,IAAM,G  
AAG,IAAK,GAAE,EAAG,GAAE,IAAM,GAAE,C;;IAEjC,OAAO,I;G;EAGX,MAAM,iBAakB,GAAE,iB;EC9C1  
B,MAAM,KAAM,GAAE,qB;IAKZ,IAAI,KAAM,GAAE,GAAl,GAAE,C;IAMIB,IAAI,MAAO,GAAE,IAAK,GA  
AE,C;G;EAGtB,MAAM,KAAK,WAAW,GAAE,OACf,OADe,cAET,MAFS,cAGV,EAHU,C;EAkBzB,MAAM,KA  
AK,UAAW,GAAE,E;EAQxB,MAAM,KAAK,QAAS,GAAE,iB;IACpB,IAAI,IAAK,IAAG,KAAM,IAAG,KAAM,  
GAAE,GAA7B,C;MACE,IAAI,YAAW,MAAM,KAAK,UAAU,CAAC,KAAD,C;MACrC,IAAI,SAAJ,C;QACE,O  
AAO,S;QAIX,IAAI,MAAM,IAAI,MAAM,KAAsC,CAAgB,KAAM,GAAE,CAAxB,EAA2B,KAAM,GAAE,CAAE  
,GAAE,EAAF,GAAO,CAA5C,C;IACV,IAAI,IAAK,IAAG,KAAM,IAAG,KAAM,GAAE,GAA7B,C;MACE,MAA  
M,KAAK,UAAU,CAAC,KAAD,CAAQ,GAAE,G;KAEjC,OAAO,G;G;EAYT,MAAM,KAAK,WAAW,GAAE,iB;I  
ACvB,IAAI,KAAK,CAAC,KAAD,CAAT,C;MACE,OAAO,MAAM,KAAK,K;WACb,IAAI,KAAM,IAAG,CAAC,  
MAAM,KAAK,gBAAzB,C;MACL,OAAO,MAAM,KAAK,U;WACb,IAAI,KAAM,GAAE,CAAE,IAAG,MAAM,  
KAAK,gBAA5B,C;MACL,OAAO,MAAM,KAAK,U;WACb,IAAI,KAAM,GAAE,CAAZ,C;MACL,OAAO,MAA  
M,KAAK,WAAW,CAAC,CAAC,KAAsC,CAAQ,OAAO,E;;MAE5C,OAAO,IAAI,MAAM,KAAsC,CACF,KAAM,  
GAAE,MAAM,KAAK,gBAAkB,GAAE,CADrC,EAEF,KAAM,GAAE,MAAM,KAAK,gBAAkB,GAAE,CAFrC,C  
;;G;EAcX,MAAM,KAAK,SAAU,GAAE,6B;IACrB,OAAO,IAAI,MAAM,KAAsC,CAAgB,OAAbB,EAAyB,QAAz  
B,C;G;EAWT,MAAM,KAAK,WAAW,GAAE,0B;IACvB,IAAI,GAAG,OAAQ,IAAG,CAAiB,C;MACE,MAAM,K  
AAK,CAAC,mCAAD,C;KAGb,IAAI,QAAQ,SAAU,IAAG,E;IACzB,IAAI,KAAM,GAAE,CAAE,IAAG,EAAG,G  
AAE,KAAiB,C;MACE,MAAM,KAAK,CAAC,sBAAuB,GAAE,KAA1B,C;KAGb,IAAI,GAAG,OAAO,CAAC,C  
AAD,CAAI,IAAG,GAArB,C;MACE,OAAO,MAAM,KAAK,WAAW,CAAC,GAAG,UAAU,CAAC,CAAD,CAAd  
,EAAmB,KAAmB,CAAyB,OAAO,E;WACxD,IAAI,GAAG,QAAQ,CAAC,GAAD,CAAM,IAAG,CAAxB,C;MAC  
L,MAAM,KAAK,CAAC,+CAAgD,GAAE,GAAnD,C;KAKb,IAAI,eAAe,MAAM,KAAK,WAAW,CAAC,IAAI,IA  
AI,CAAC,KAAD,EAAQ,CAAR,CAAT,C;IAEzC,IAAI,SAAS,MAAM,KAAK,K;IACxB,KAAK,IAAI,IAAI,CAA  
b,EAAgB,CAAE,GAAE,GAAG,OAAvB,EAAgC,CAAE,IAAG,CAArC,C;MACE,IAAI,OAAO,IAAI,IAAI,CAAC  
,CAAD,EAAl,GAAG,OAAQ,GAAE,CAAjB,C;MACnB,IAAI,QAAQ,QAAQ,CAAC,GAAG,UAAU,CAAC,CAA  
D,EAAl,CAAE,GAAE,IAAR,CAAd,EAA6B,KAA7B,C;MACpB,IAAI,IAAK,GAAE,CAAX,C;QACE,IAAI,QAA  
Q,MAAM,KAAK,WAAW,CAAC,IAAI,IAAI,CAAC,KAAD,EAAQ,IAAR,CAAT,C;QACIC,MAAO,GAAE,MAA  
M,SAAS,CAAC,KAAD,CAAO,IAAI,CAAC,MAAM,KAAK,WAAW,CAAC,KAAD,CAAvB,C;;QAEnC,MAAO,  
GAAE,MAAM,SAAS,CAAC,YAAD,C;QACxB,MAAO,GAAE,MAAM,IAAI,CAAC,MAAM,KAAK,WAAW,CA  
AC,KAAD,CAAvB,C;;IAGvB,OAAO,M;G;EAcT,MAAM,KAAK,gBAAiB,GAAE,CAAE,IAAG,E;EAOnC,MA  
AM,KAAK,gBAAiB,GAAE,CAAE,IAAG,E;EAOnC,MAAM,KAAK,gBAAiB,GACxB,MAAM,KAAK,gBAAiB,  
GAAE,MAAM,KAAK,gB;EA07C,MAAM,KAAK,gBAAiB,GACxB,MAAM,KAAK,gBAAiB,GAAE,C;EA0IC,  
MAAM,KAAK,gBAAiB,GACxB,MAAM,KAAK,gBAAiB,GAAE,MAAM,KAAK,gB;EA07C,MAAM,KAAK,gB  
AAiB,GACxB,MAAM,KAAK,gBAAiB,GAAE,MAAM,KAAK,gB;EA07C,MAAM,KAAK,gBAAiB,GACxB,MA  
AM,KAAK,gBAAiB,GAAE,C;EAIIC,MAAM,KAAK,KAAM,GAAE,MAAM,KAAK,QAAQ,CAAC,CAAD,C;EA  
IIC,MAAM,KAAK,IAAK,GAAE,MAAM,KAAK,QAAQ,CAAC,CAAD,C;EAIrC,MAAM,KAAK,QAAS,GAAE,  
MAAM,KAAK,QAAQ,CAAC,EAAD,C;EAIzC,MAAM,KAAK,UAAW,GACIB,MAAM,KAAK,SAAS,CAAC,aA  
AW,GAAE,CAAd,EAAiB,UAAW,GAAE,CAA9B,C;EAIxB,MAAM,KAAK,UAAW,GAAE,MAAM,KAAK,SA  
AS,CAAC,CAAD,EAAl,aAAW,GAAE,CAAjB,C;EA05C,MAAM,KAAK,YAAa,GAAE,MAAM,KAAK,QAAQ,C  
AAC,CAAE,IAAG,EAAN,C;EAI7C,MAAM,KAAK,UAAU,MAAO,GAAE,Y;IAC5B,OAAO,IAAI,K;G;EAKb,M  
AAM,KAAK,UAAU,SAAU,GAAE,Y;IAC/B,OAAO,IAAI,MAAO,GAAE,MAAM,KAAK,gBAAiB,GACzC,IAAI

,mBAAmB,E;G;EAIhC,MAAM,KAAK,UAAU,SAAU,GAAE,Y;IAC/B,OAAO,IAAI,MAAO,GAAE,IAAI,K;G;E  
AQ1B,MAAM,KAAK,UAAU,SAAU,GAAE,qB;IAC/B,IAAI,QAAQ,SAAU,IAAG,E;IACzB,IAAI,KAAM,GAAE  
,CAAE,IAAG,EAAG,GAAE,KAAtB,C;MACE,MAAM,KAAK,CAAC,sBAAuB,GAAE,KAA1B,C;KAGb,IAAI,I  
AAI,OAAO,EAAf,C;MACE,OAAO,G;KAGT,IAAI,IAAI,WAAW,EAAnB,C;MACE,IAAI,IAAI,WAAW,CAAC,  
MAAM,KAAK,UAAZ,CAAnB,C;QAGE,IAAI,YAA Y,MAAM,KAAK,WAAW,CAAC,KAAD,C;QACtC,IAAI,M  
AAM,IAAI,IAAI,CAAC,SAAD,C;QACIB,IAAI,MAAM,GAAG,SAAS,CAAC,SAAD,CAAW,SAAS,CAAC,IAA  
D,C;QAC1C,OAAO,GAAG,SAAS,CAAC,KAAD,CAAQ,GAAE,GAAG,MAAM,EAAE,SAAS,CAAC,KAAD,C;;  
QAEjD,OAAO,GAAI,GAAE,IAAI,OAAO,EAAE,SAAS,CAAC,KAAD,C;;KAMvC,IAAI,eAAe,MAAM,KAAK,  
WAAW,CAAC,IAAI,IAAI,CAAC,KAAD,EAAQ,CAAR,CAAT,C;IAEzC,IAAI,MAAM,I;IACV,IAAI,SAAS,E;IA  
Cb,OAAO,IAAP,C;MACE,IAAI,SAAS,GAAG,IAAI,CAAC,YAAD,C;MACpB,IAAI,SAAS,GAAG,SAAS,CAAC  
,MAAM,SAAS,CAAC,YAAD,CAAhB,CAA+B,MAAM,E;MAC9D,IAAI,SAAS,MAAM,SAAS,CAAC,KAAD,C;  
MAE5B,GAAI,GAAE,M;MACN,IAAI,GAAG,OAAO,EAAd,C;QACE,OAAO,MAAO,GAAE,M;;QAEhB,OAAO,  
MAAM,OAAQ,GAAE,CAA vB,C;UACE,MAAO,GAAE,GAAI,GAAE,M;;QAEjB,MAAO,GAAE,EAAG,GAAE,  
MAAO,GAAE,M;;G;EAO7B,MAAM,KAAK,UAAU,YAAa,GAAE,Y;IACIC,OAAO,IAAI,M;G;EAKb,MAAM,K  
AAK,UAAU,WAA Y,GAAE,Y;IACjC,OAAO,IAAI,K;G;EAKb,MAAM,KAAK,UAAU,mBAAoB,GAAE,Y;IACz  
C,OAAQ,IAAI,KAAM,IAAG,CAAG,GACpB,IAAI,KADgB,GACR,MAAM,KAAK,gBAAiB,GAAE,IAAI,K;G;E  
AQpD,MAAM,KAAK,UAAU,cAAe,GAAE,Y;IACpC,IAAI,IAAI,WAAW,EAAnB,C;MACE,IAAI,IAAI,WAAW,  
CAAC,MAAM,KAAK,UAAZ,CAAnB,C;QACE,OAAO,E;;QAEp,OAAO,IAAI,OAAO,EAAE,cAAc,E;;MAGpC,  
IAAI,MAAM,IAAI,MAAO,IAAG,CAAE,GAAE,IAAI,MAAN,GAAe,IAAI,K;MAC7C,KAAK,IAAI,MAAM,EA  
Af,EAAmB,GAAI,GAAE,CAAzB,EAA4B,GAAG,EAA/B,C;QACE,IAAuB,CAA1B,GAAI,GAAG,CAAE,IAAG,  
GAAM,KAAG,CAA1B,C;UACE,K;;MAGJ,OAAO,IAAI,MAAO,IAAG,CAAE,GAAE,GAAI,GAAE,EAAR,GAA  
a,GAAI,GAAE,C;;G;EAM9C,MAAM,KAAK,UAAU,OAAQ,GAAE,Y;IAC7B,OAAO,IAAI,MAAO,IAAG,CAAE  
,IAAG,IAAI,KAAM,IAAG,C;G;EAKzC,MAAM,KAAK,UAAU,WAA Y,GAAE,Y;IACjC,OAAO,IAAI,MAAO,G  
AAE,C;G;EAKtB,MAAM,KAAK,UAAU,MAAO,GAAE,Y;IAC5B,OAAuB,CAAf,IAAI,KAAM,GAAE,CAAG,K  
AAG,C;G;EAQ5B,MAAM,KAAK,UAAU,WAA Y,GAAE,iB;IACjC,OAAQ,IAAI,MAAO,IAAG,KAAK,MAAQ,I  
AAI,IAAI,KAAM,IAAG,KAAK,K;G;EAQ3D,MAAM,KAAK,UAAU,cAAe,GAAE,iB;IACpC,OAAQ,IAAI,MAA  
O,IAAG,KAAK,MAAQ,IAAI,IAAI,KAAM,IAAG,KAAK,K;G;EAQ3D,MAAM,KAAK,UAAU,SAAU,GAAE,iB;  
IAC/B,OAAO,IAAI,QAAQ,CAAC,KAAD,CAAQ,GAAE,C;G;EAQ/B,MAAM,KAAK,UAAU,gBAAiB,GAAE,iB  
;IACtC,OAAO,IAAI,QAAQ,CAAC,KAAD,CAAQ,IAAG,C;G;EAQhC,MAAM,KAAK,UAAU,YAAa,GAAE,iB;I  
ACIC,OAAO,IAAI,QAAQ,CAAC,KAAD,CAAQ,GAAE,C;G;EAQ/B,MAAM,KAAK,UAAU,mBAAoB,GAAE,iB  
;IACzC,OAAO,IAAI,QAAQ,CAAC,KAAD,CAAQ,IAAG,C;G;EAUhC,MAAM,KAAK,UAAU,QAAS,GAAE,iB;I  
AC9B,IAAI,IAAI,WAAW,CAAC,KAAD,CAAnB,C;MACE,OAAO,C;KAGT,IAAI,UAAU,IAAI,WAAW,E;IAC7  
B,IAAI,WAAW,KAAK,WAAW,E;IAC/B,IAAI,OAAQ,IAAG,CAAC,QAAhB,C;MACE,OAAO,E;KAET,IAAI,C  
AAC,OAAQ,IAAG,QAAhB,C;MACE,OAAO,C;KAIT,IAAI,IAAI,SAAS,CAAC,KAAD,CAAO,WAAW,EAAnC,  
C;MACE,OAAO,E;;MAEP,OAAO,C;;G;EAMX,MAAM,KAAK,UAAU,OAAQ,GAAE,Y;IAC7B,IAAI,IAAI,WA  
AW,CAAC,MAAM,KAAK,UAAZ,CAAnB,C;MACE,OAAO,MAAM,KAAK,U;;MAEIB,OAAO,IAAI,IAAI,EAA  
E,IAAI,CAAC,MAAM,KAAK,IAAZ,C;;G;EAUzB,MAAM,KAAK,UAAU,IAAK,GAAE,iB;IAG1B,IAAI,MAAM  
,IAAI,MAAO,KAAI,E;IACzB,IAAI,MAAM,IAAI,MAAO,GAAE,K;IACvB,IAAI,MAAM,IAAI,KAAM,KAAI,E;I  
ACxB,IAAI,MAAM,IAAI,KAAM,GAAE,K;IAEtB,IAAI,MAAM,KAAK,MAAO,KAAI,E;IAC1B,IAAI,MAAM,K  
AAK,MAAO,GAAE,K;IACxB,IAAI,MAAM,KAAK,KAAM,KAAI,E;IACzB,IAAI,MAAM,KAAK,KAAM,GAA  
E,K;IAEvB,IAAI,MAAM,CAAV,EAAa,MAAM,CAAnB,EAA sB,MAAM,CAA5B,EAA+B,MAAM,C;IACrC,GA  
AI,IAAG,GAAI,GAAE,G;IACb,GAAI,IAAG,GAAI,KAAI,E;IACf,GAAI,IAAG,K;IACP,GAAI,IAAG,GAAI,GA  
AE,G;IACb,GAAI,IAAG,GAAI,KAAI,E;IACf,GAAI,IAAG,K;IACP,GAAI,IAAG,GAAI,GAAE,G;IACb,GAAI,I  
AAG,GAAI,KAAI,E;IACf,GAAI,IAAG,K;IACP,GAAI,IAAG,GAAI,GAAE,G;IACb,GAAI,IAAG,K;IACP,OAAO  
,MAAM,KAAK,SAAS,CAAE,GAAI,IAAG,EAAL,GAAE,GA Af,EAAqB,GAAI,IAAG,EAAL,GAAE,GAALC,C;G;  
EAS7B,MAAM,KAAK,UAAU,SAAU,GAAE,iB;IAC/B,OAAO,IAAI,IAAI,CAAC,KAAK,OAAO,EAAb,C;G;EA  
SjB,MAAM,KAAK,UAAU,SAAU,GAAE,iB;IAC/B,IAAI,IAAI,OAAO,EA Af,C;MACE,OAAO,MAAM,KAAK,K  
;WACb,IAAI,KAAK,OAAO,EA AhB,C;MACL,OAAO,MAAM,KAAK,K;KAGpB,IAAI,IAAI,WAAW,CAAC,M

AAM,KAAK,UAAZ,CAAnB,C;MACE,OAAO,KAAK,MAAM,EAAG,GAAE,MAAM,KAAK,UAAb,GAA0B,M  
AAM,KAAK,K;WACrD,IAAI,KAAK,WAAW,CAAC,MAAM,KAAK,UAAZ,CAApB,C;MACL,OAAO,IAAI,M  
AAM,EAAG,GAAE,MAAM,KAAK,UAAb,GAA0B,MAAM,KAAK,K;KAG3D,IAAI,IAAI,WAAW,EAAnB,C;M  
ACE,IAAI,KAAK,WAAW,EAAPB,C;QACE,OAAO,IAAI,OAAO,EAAE,SAAS,CAAC,KAAK,OAAO,EAAb,C;;  
QAE7B,OAAO,IAAI,OAAO,EAAE,SAAS,CAAC,KAAD,CAAO,OAAO,E;;WAExC,IAAI,KAAK,WAAW,EAAP  
B,C;MACL,OAAO,IAAI,SAAS,CAAC,KAAK,OAAO,EAAb,CAAgB,OAAO,E;KAI7C,IAAI,IAAI,SAAS,CAAC,  
MAAM,KAAK,YAAZ,CAA0B,IACvC,KAAK,SAAS,CAAC,MAAM,KAAK,YAAZ,CADIB,C;MAEE,OAAO,M  
AAM,KAAK,WAAW,CAAC,IAAI,SAAS,EAAG,GAAE,KAAK,SAAS,EAajC,C;KAM/B,IAAI,MAAM,IAAI,M  
AAO,KAAI,E;IACzB,IAAI,MAAM,IAAI,MAAO,GAAE,K;IACvB,IAAI,MAAM,IAAI,KAAM,KAAI,E;IACxB,I  
AAI,MAAM,IAAI,KAAM,GAAE,K;IAEtB,IAAI,MAAM,KAAK,MAAO,KAAI,E;IAC1B,IAAI,MAAM,KAAK,  
MAAO,GAAE,K;IACxB,IAAI,MAAM,KAAK,KAAM,KAAI,E;IACzB,IAAI,MAAM,KAAK,KAAM,GAAE,K;I  
AEvB,IAAI,MAAM,CAAV,EAAa,MAAM,CAAnB,EAAsB,MAAM,CAA5B,EAA+B,MAAM,C;IACrC,GAAI,IA  
AG,GAAI,GAAE,G;IACb,GAAI,IAAG,GAAI,KAAI,E;IACf,GAAI,IAAG,K;IACP,GAAI,IAAG,GAAI,GAAE,G;I  
ACb,GAAI,IAAG,GAAI,KAAI,E;IACf,GAAI,IAAG,K;IACP,GAAI,IAAG,GAAI,GAAE,G;IACb,GAAI,IAAG,G  
AAI,KAAI,E;IACf,GAAI,IAAG,K;IACP,GAAI,IAAG,GAAI,GAAE,G;IACb,GAAI,IAAG,GAAI,KAAI,E;IACf,G  
AAI,IAAG,K;IACP,GAAI,IAAG,GAAI,GAAE,G;IACb,GAAI,IAAG,GAAI,KAAI,E;IACf,GAAI,IAAG,K;IACP,  
GAAI,IAAG,GAAI,GAAE,G;IACb,GAAI,IAAG,GAAI,KAAI,E;IACf,GAAI,IAAG,K;IACP,GAAI,IAAG,GAAI,  
GAAE,GAAI,GAAE,GAAI,GAAE,GAAI,GAAE,GAAI,GAAE,GAAI,GAAE,GAAI,GAAE,G;IACjD,GAAI,IAA  
G,K;IACP,OAAO,MAAM,KAAK,SAAS,CAAE,GAAI,IAAG,EAAl,GAAE,GAaf,EAaqB,GAAI,IAAG,EAAl,G  
AAE,GAAl,C;G;EAS7B,MAAM,KAAK,UAAU,IAAK,GAAE,iB;IAC1B,IAAI,KAAK,OAAO,EAAb,C;MACE  
,MAAM,KAAK,CAAC,kBAAD,C;WACN,IAAI,IAAI,OAAO,EAaf,C;MACL,OAAO,MAAM,KAAK,K;KAGpB,  
IAAI,IAAI,WAAW,CAAC,MAAM,KAAK,UAAZ,CAAnB,C;MACE,IAAI,KAAK,WAAW,CAAC,MAAM,KAA  
K,IAAZ,CAAkB,IACIC,KAAK,WAAW,CAAC,MAAM,KAAK,QAaz,CADpB,C;QAEe,OAAO,MAAM,KAAK,  
U;aACb,IAAI,KAAK,WAAW,CAAC,MAAM,KAAK,UAAZ,CAApB,C;QACL,OAAO,MAAM,KAAK,I;;QAGIB,  
IAAI,WAAW,IAAI,WAAW,CAAC,CAAD,C;QAC9B,IAAI,SAAS,QAAQ,IAAI,CAAC,KAAD,CAAO,UAAU,C  
AAC,CAAD,C;QAC1C,IAAI,MAAM,WAAW,CAAC,MAAM,KAAK,KAAZ,CAArB,C;UACE,OAAO,KAAK,W  
AAW,EAAG,GAAE,MAAM,KAAK,IAAb,GAAoB,MAAM,KAAK,Q;;UAEzD,IAAI,MAAM,IAAI,SAAS,CAAC,  
KAAK,SAAS,CAAC,MAAD,CAaf,C;UACvB,IAAI,SAAS,MAAM,IAAI,CAAC,GAAG,IAAI,CAAC,KAAD,CA  
AR,C;UACvB,OAAO,M;;;WAGN,IAAI,KAAK,WAAW,CAAC,MAAM,KAAK,UAAZ,CAApB,C;MACL,OAAO  
,MAAM,KAAK,K;KAGpB,IAAI,IAAI,WAAW,EAAnB,C;MACE,IAAI,KAAK,WAAW,EAAPB,C;QACE,OAAO  
,IAAI,OAAO,EAAE,IAAI,CAAC,KAAK,OAAO,EAAb,C;;QAExB,OAAO,IAAI,OAAO,EAAE,IAAI,CAAC,KA  
AD,CAAO,OAAO,E;;WAEEnC,IAAI,KAAK,WAAW,EAAPB,C;MACL,OAAO,IAAI,IAAI,CAAC,KAAK,OAAO,  
EAAb,CAAgB,OAAO,E;KAQxC,IAAI,MAAM,MAAM,KAAK,K;IACrB,IAAI,MAAM,I;IACV,OAAO,GAAG,m  
BAAmB,CAAC,KAAD,CAA7B,C;MAGE,IAAI,SAAS,IAAI,IAAI,CAAC,CAAD,EAAl,IAAI,MAAM,CAAC,GA  
AG,SAAS,EAAG,GAAE,KAAK,SAAS,EAAbC,CAAd,C;MAIrB,IAAI,OAAO,IAAI,KAAK,CAAC,IAAI,IAAI,C  
AAC,MAAD,CAAS,GAAE,IAAI,IAAxB,C;MACpB,IAAI,QAAS,IAAK,IAAG,EAAl,GAAE,CAAF,GAAM,IAAI  
,IAAI,CAAC,CAAD,EAAl,IAAK,GAAE,EAAX,C;MAIvC,IAAI,YAAY,MAAM,KAAK,WAAW,CAAC,MAAD,  
C;MACtC,IAAI,YAAY,SAAS,SAAS,CAAC,KAAD,C;MACIC,OAAO,SAAS,WAAW,EAAG,IAAG,SAAS,YAA  
Y,CAAC,GAAD,CAAtD,C;QACE,MAAO,IAAG,K;QACV,SAAU,GAAE,MAAM,KAAK,WAAW,CAAC,MAA  
D,C;QACIC,SAAU,GAAE,SAAS,SAAS,CAAC,KAAD,C;;MAKhC,IAAI,SAAS,OAAO,EAAPB,C;QACE,SAAU,  
GAAE,MAAM,KAAK,I;OAGzB,GAAI,GAAE,GAAG,IAAI,CAAC,SAAD,C;MACb,GAAI,GAAE,GAAG,SAAS,  
CAAC,SAAD,C;;IAEPb,OAAO,G;G;EAST,MAAM,KAAK,UAAU,OAAQ,GAAE,iB;IAC7B,OAAO,IAAI,SAAS,  
CAAC,IAAI,IAAI,CAAC,KAAD,CAAO,SAAS,CAAC,KAAD,CAAzB,C;G;EAKtB,MAAM,KAAK,UAAU,IAA  
K,GAAE,Y;IAC1B,OAAO,MAAM,KAAK,SAAS,CAAC,CAAC,IAAI,KAAI,EAaa,CAAC,IAAI,MAAI,B,C;G;E  
AS7B,MAAM,KAAK,UAAU,IAAK,GAAE,iB;IAC1B,OAAO,MAAM,KAAK,SAAS,CAAC,IAAI,KAAM,GAAE  
,KAAK,KAAIB,EACI,IAAI,MAAO,GAAE,KAAK,MADtB,C;G;EAU7B,MAAM,KAAK,UAAU,GAAI,GAAE,iB  
;IACzB,OAAO,MAAM,KAAK,SAAS,CAAC,IAAI,KAAM,GAAE,KAAK,KAAIB,EACI,IAAI,MAAO,GAAE,K  
AAK,MADtB,C;G;EAU7B,MAAM,KAAK,UAAU,IAAK,GAAE,iB;IAC1B,OAAO,MAAM,KAAK,SAAS,CAAC

,IAAI,KAAM,GAAE,KAAK,KAAIB,EACI,IAAI,MAAO,GAAE,KAAK,MADtB,C;G;EAU7B,MAAM,KAAK,UAAU,UAAW,GAAE,mB;IACHc,OAAQ,IAAG,E;IACX,IAAI,OAAQ,IAAG,CAAf,C;MACE,OAAO,I;MAEP,IAAI,MAAM,IAAI,K;MACd,IAAI,OAAQ,GAAE,EAAd,C;QACE,IAAI,OAAO,IAAI,M;QACf,OAAO,MAAM,KAAK,SAAS,CACvB,GAAl,IAAG,OADgB,EAEtB,IAAK,IAAG,OAAS,GAAG,GAAl,KAAK,EAAG,GAAE,OAFZ,C;QAI3B,OAAO,MAAM,KAAK,SAAS,CAAC,CAAD,EAAl,GAAl,IAAI,OAAQ,GAAE,EAAtB,C;G;EAWjC,MAAM,KAAK,UAAU,WAAY,GAAE,mB;IACjC,OAAQ,IAAG,E;IACX,IAAI,OAAQ,IAAG,CAAf,C;MACE,OAAO,I;MAEP,IAAI,OAAO,IAAI,M;MACf,IAAI,OAAQ,GAAE,EAAd,C;QACE,IAAI,MAAM,IAAI,K;QACd,OAAO,MAAM,KAAK,SAAS,CACtB,GAAl,KAAI,OAAS,GAAG,IAAK,IAAI,EAAG,GAAE,OADZ,EAEvB,IAAK,IAAG,OAFc,C;QAI3B,OAAO,MAAM,KAAK,SAAS,CACvB,IAAK,IAAI,OAAQ,GAAE,EADI,EAEvB,IAAK,IAAG,CAAE,GAAE,CAAF,GAAM,EAFO,C;G;EAejC,MAAM,KAAK,UAAU,mBAaOb,GAAE,mB;IACzC,OAAQ,IAAG,E;IACX,IAAI,OAAQ,IAAG,CAAf,C;MACE,OAAO,I;MAEP,IAAI,OAAO,IAAI,M;MACf,IAAI,OAAQ,GAAE,EAAd,C;QACE,IAAI,MAAM,IAAI,K;QACd,OAAO,MAAM,KAAK,SAAS,CACtB,GAAl,KAAI,OAAS,GAAG,IAAK,IAAI,EAAG,GAAE,OADZ,EAEvB,IAAK,KAAI,OAFc,C;aAGtB,IAAI,OAAQ,IAAG,EAaf,C;QACL,OAAO,MAAM,KAAK,SAAS,CAAC,IAAD,EAAO,CAAP,C;QAE3B,OAAO,MAAM,KAAK,SAAS,CAAC,IAAK,KAAK,OAAQ,GAAE,EAARb,EAAOb,CAAIB,C;G;EAMjC,MAAM,KAAK,UAAU,OAAQ,GAAE,iB;IAC3B,OAAO,KAAM,YAAW,MAAM,KAAM,IAAG,IAAI,WAAY,CAAC,KAAD,C;G;EAG1D,MAAM,KAAK,UAAU,gBAaIB,GAAE,MAAM,KAAK,UAAU,Q;EAE7D,MAAM,KAAK,UAAU,IAAK,GAAE,Y;IACxB,OAAO,IAAI,IAAI,CAAC,MAAM,KAAK,IAAZ,C;G;EAGnB,MAAM,KAAK,UAAU,IAAK,GAAE,Y;IACxB,OAAO,IAAI,IAAI,CAAC,MAAM,KAAK,QAAs,C;G;EAGnB,MAAM,KAAK,UAAU,QAAS,GAAE,Y;IAC5B,OAAO,IAAI,SAAS,E;G;EAGxB,MAAM,KAAK,UAAU,UAAW,GAAE,Y;IAC9B,OAAO,I;G;EAGX,MAAM,KAAK,UAAU,WAAY,GAAE,MAAM,KAAK,UAAU,O;EACxD,MAAM,KAAK,UAAU,IAAK,GAAE,MAAM,KAAK,UAAU,I;EAEjD,MAAM,KAAK,UAAU,QAAS,GAAE,iB;IAC5B,OAAO,IAAI,MAAM,OAAO,OAAO,UAAxB,CAAmC,IAAnC,EAAYc,KAAzC,C;G;EC1zBX,MAAM,aAAc,GAAE,2B;G;EAGtB,MAAM,qBAAsB,GAAE,oB;IAC1B,OAAO,G;G;EAGX,MAAM,aAAc,GAAE,e;IACIB,IAAI,IAAI,Y;MACJ,CAAE,GAAE,GAAG,E;MACP,OAAO,CAAC,MAAM,CAAC,IAAD,EAAO,SAAP,C;K;IAEIB,OAAO,Y;MACH,OAAO,CAAC,MAAM,CAAC,IAAD,EAAO,SAAP,C;K;G;EAItB,MAAM,SAAU,GAAE,gB;IACd,OAAO,kB;MACH,OAAO,OAAO,MAAO,KAAI,I;K;G;EAIjC,MAAM,aAAc,GAAE,iB;IACIB,OAAO,kB;MACH,OAAO,MAAM,OAAO,CAAC,MAAD,EAAS,KAAT,C;K;G;EAI5B,MAAM,OAAQ,GAAE,c;IACZ,OAAO,kB;MACH,OAAO,MAAO,IAAG,IAAK,IAAG,EAEE,CAAC,MAAD,C;K;G;EAIcC,MAAM,aAAc,GAAE,gB;IACIB,OAAO,kB;MACH,OAAO,CAAC,CAAC,MAAD,CAAS,IAAG,CAAC,CAAC,MAAD,C;K;G;EAI7B,MAAM,qBAAsB,GAAE,wC;G;EAG9B,MAAM,YAAa,GAAE,iB;IACjB,OAAO,K;G;EAGX,MAAM,gBAaIB,GAAE,qB;IACrB,gBAAgB,E;G;EAGpB,MAAM,oBAaQb,GAAE,qB;IACzB,gBAAgB,E;G;EAGpB,MAAM,kBAAmB,GAAE,qB;IACvB,gBAAgB,E;G;EAGpB,MAAM,mBAaOb,GAAE,4B;IACxB,gBAAgB,E;G;EAGpB,MAAM,6BAa8B,GAAE,yB;IACIC,gBAAgB,E;G;EAGpB,4B;IACI,MAAM,IAAI,KAAJ,CACf,iDAaKD,GACID,qDAAsD,GACtD,uDAHE,C;G;EAMV,MAAM,gBAaIB,GAAE,4B;IACrB,OAAO,Y;MACH,OAAO,Y;K;G;ECJfF,MAAM,UAAW,GAAE,gB;IACf,IAAI,QAAQ,OAAO,C;IACnB,IAAI,KAAM,KAAI,QAAAd,C;MACI,IAAI,OAAO,CAAE,KAAI,QAAjB,C;QACI,OAAO,MAAM,gBAAgB,CAAC,CAAD,EAAl,CAAJ,C;OAEjC,OAAO,MAAM,mBAAmB,CAAC,CAAD,EAAl,CAAJ,C;KAEpC,IAAI,KAAM,KAAI,QAAS,IAAG,KAAAM,KAAI,SAAP,C;MACI,OAAO,MAAM,mBAAmB,CAAC,CAAD,EAAl,CAAJ,C;KAEpC,OAAO,CAAC,gBAAgB,CAAC,CAAD,C;G;EAG5B,MAAM,mBAaOb,GAAE,gB;IACxB,OAAO,CAAE,GAAE,CAAE,GAAE,EAaf,GAAO,CAAE,GAAE,CAAE,GAAE,CAAF,GAAM,C;G;EAGpC,MAAM,gBAaIB,GAAE,gB;IACrB,IAAI,CAAE,GAAE,CAAR,C;MAAW,OAAO,E;IACIB,IAAI,CAAE,GAAE,CAAR,C;MAAW,OAAO,C;IAEIB,IAAI,CAAE,KAAI,CAAV,C;MACI,IAAI,CAAE,KAAI,CAAV,C;QAAa,OAAO,C;MAEPB,IAAI,KAAK,CAAE,GAAE,C;MACb,OAAO,EAAG,KAAI,CAAE,GAAE,CAAE,GAAE,CAAF,GAAG,EAAG,GAAE,CAAE,GAAE,EAaf,GAAG,C;KAG7C,OAAO,CAAE,KAAI,CAAE,GAAG,CAAE,KAAI,CAAE,GAAE,CAAF,GAAM,CAAJB,GAAsB,E;G;EAGzC,MAAM,QAAS,GAAE,iB;IACb,OAAO,MAAM,OAAO,CAAC,KAAK,GAAC,CAAP,C;G;EAGxB,MAAM,QAAS,GAAE,iB;IACb,OAAO,MAAM,OAAO,CAAC,KAAK,GAAC,CAAP,C;G;EAGxB,MAAM,KAAM,GAAE,IAAI,KAAM,IAAG,I;EAE3B,MAAM,aAAc,GAAE,I;EAEtB,oB;IACI,OAAyB,CAAhB,CAAE,GAAE,YAAY,KAAg,CAAE,GAAE,KAAp,CAAE,GAAE,CAAz,CAAE,GAAE,KAAQ,KAAg,CAAE,GAAE,CAAP,CAAW

,GAAE,C;G;EA6DtE,CA1DD,Y;IACG,IAAI,MAAM,IAAI,WAAJ,CAAkB,CAAhB,C;IACV,IAAI,aAAa,IAAI,YAAJ,CAAiB,GAAjB,C;IACjB,IAAI,aAAa,IAAI,YAAJ,CAAiB,GAAjB,C;IACjB,IAAI,WAAW,IAAI,UAAJ,CAAe,GAAf,C;IACf,IAAI,WAAW,C;IACf,IAAI,YAAJ,C;IAEhB,UAAU,CAAC,CAAD,CAAI,GAAE,E;IACbB,IAAI,QAAQ,CAAC,QAAD,CAAW,KAAI,CAA3B,C;MACI,QAAS,GAAE,C;MACX,SAAU,GAAE,C;KAGhB,MAAM,aAAc,GAAE,iB;MACiB,OAAO,MAAM,gBAAgB,CAAC,KAAK,CAAC,KAAD,CAAQ,GAAE,GAAF,GAAQ,KAAtB,C;K;IAGjC,MAAM,gBAAiB,GAAE,iB;MACrB,UAAU,CAAC,CAAD,CAAI,GAAE,K;MACHb,OAAO,MAAM,KAAK,SAAS,CAAC,QAAQ,CAAC,QAAD,CAAT,EAAqB,QAAQ,CAAC,SAAD,CAA7B,C;K;IAG/B,MAAM,eAAgB,GAAE,iB;MACpB,QAAQ,CAAC,QAAD,CAAW,GAAE,KAAK,K;MAC1B,QAAQ,CAAC,SAAD,CAAAY,GAAE,KAAK,M;MAC3B,OAAO,UAAU,CAAC,CAAD,C;K;IAGrB,MAAM,YAAa,GAAE,iB;MACjB,OAAO,MAAM,eAAe,CAAC,KAAK,CAAC,KAAD,CAAQ,GAAE,GAAF,GAAQ,KAAtB,C;K;IAGhC,MAAM,eAAgB,GAAE,iB;MACpB,UAAU,CAAC,CAAD,CAAI,GAAE,K;MACHb,OAAO,QAAQ,CAAC,CAAD,C;K;IAGnB,MAAM,cAAe,GAAE,iB;MACnB,QAAQ,CAAC,CAAD,CAAI,GAAE,K;MACd,OAAO,UAAU,CAAC,CAAD,C;K;IAIrB,MAAM,cAAe,GAAE,iB;MACnB,UAAU,CAAC,CAAD,CAAI,GAAE,K;MACHb,OAAO,QAAQ,CAAC,SAAD,CAAY,GAAE,a;K;IAGjC,MAAM,eAAgB,GAAE,e;MACpB,IAAc,CAAT,GAAl,GAAE,CAAG,MAAI,GAAlB,C;QACI,OAAO,GAAl,GAAE,C;;QAGb,UAAU,CAAC,CAAD,CAAI,GAAE,G;QACHb,OAAc,CAA9B,QAAQ,CAAC,SAAD,CAAY,GAAE,EAAG,GAAE,CAAG,IAAE,QAAQ,CAAC,QAAD,CAAW,GAAE,C;;K;GAGvE,G;EAEF,MAAM,cAAe,GAAE,a;IACnB,OAAO,CAAE,IAAG,IAAK,GAAE,CAAF,GAAM,MAAM,SAAS,E;G;EC7G1C,IAAI,OAAO,MAAM,UAAU,WAAJ,KAAl,WAA3C,C;IACI,MAAM,eAAe,CAAC,MAAM,UAAp,EAAmB,YAAAnB,EAAlC,QAC3C,kC;MACH,QAAS,GAAE,QAAS,IAAG,C;MACvB,OAAO,IAAI,YAAJ,CAAC,YAAD,EA Ae,QA Af,CAAYB,KAAl,Q;KAHN,CAAjC,C;GAOzB,IAAI,OAAO,MAAM,UAAU,SAAU,KAAl,WAAzC,C;IACI,MAAM,eAAe,CAAC,MAAM,UAAp,EAAmB,YAAAnB,EA A+B,QACzC,kC;MACH,IAAI,gBAAgB,IAAI,SAAS,E;MACjC,IAAI,QAAS,KAAl,SAAU,IAAG,QAAS,GAAE,aAAa,OAAtD,C;QACI,QAAS,GAAE,aAAa,O;OAE5B,QAAS,IAAG,YAAJ,O;MACxB,IAAI,YAAJ,aAAa,QAAQ,CAAC,YAAD,EA Ae,QA Af,C;MACrC,OAAO,SAAU,KAAl,EAAG,IAAG,SAAU,KAAl,Q;KARG,CAA/B,C;GAazB,IAAI,OAAO,IAAI,KAAM,KAAl,WAAzB,C;IACI,IAAI,KAAM,GAAE,a;MACR,CAAE,GAAE,CAAC,C;MACL,IAAI,CAAE,KAAl,CAAE,IAAG,KAAK,CAAC,CAAD,CAApB,C;QACI,OAAO,MAAM,CAAC,CAAD,C;OAEjB,OAAO,CAAE,GAAE,CAAE,GAAE,CAAF,GAAM,E;K;GAG3B,IAAI,OAAO,IAAI,MAAO,KAAl,WAA1B,C;IACI,IAAI,MAAO,GAAE,a;MACT,IAAI,KAAK,CAAC,CAAD,CAAT,C;QACI,OAAO,G;OAEX,IAAI,CAAE,GAAE,CAAR,C;QACI,OAAO,IAAI,MAAM,CAAC,CAAD,C;OAErB,OAAO,IAAI,KAAK,CAAC,CAAD,C;K;GAuKtB,CAnKD,Y;IACG,IAAI,UAAU,qB;IACd,IAAI,iBAAiB,IAAI,KAAK,CAAC,OAAD,C;IAC9B,IAAI,iBAAiB,IAAI,KAAK,CAAC,cAAD,C;IAC9B,IAAI,uBAAuB,CAAC,GAAC,c;IAC7B,IAAI,uBAAuB,CAAC,GAAC,c;IAE7B,IAAI,OAAO,IAAI,KAAM,KAAl,WAAzB,C;MACI,IAAI,KAAM,GAAE,a;QACR,IAAI,IAAI,IAAI,CAAC,CAAD,CAAI,GAAE,cAAIB,C;UACI,IAAI,SAAS,C;UACb,IAAI,IAAI,IAAI,CAAC,CAAD,CAAI,GAAE,cAAIB,C;YACI,MAAO,IAAI,CAAE,GAAE,CAAE,GAAE,CAAG,GAAE,C;WAE5B,OAAO,M;;UAEP,IAAI,IAAI,IAAI,IAAI,CAAC,CAAD,C;UACHb,IAAI,KAAK,CAAE,GAAE,C;UACb,IAAI,CAAC,QAAQ,CAAC,CAAD,CAAb,C;YAAkB,OAAO,IAAI,IAAI,CAAC,CAAE,GAAE,IAAI,IAAT,C;UACjC,IAAI,CAAC,QAAQ,CAAC,EAAD,CAAb,C;YAAmB,OAAO,CAAC,IAAI,IAAI,CAAC,CAAC,CAAE,GAAE,IAAI,IAAV,C;UACnC,OAAgB,CAAR,CAAE,GAAE,EAAl,IAAE,C;;O;KAI9B,IAAI,OAAO,IAAI,KAAM,KAAl,WAAzB,C;MACI,IAAI,KAAM,GAAE,a;QACR,IAAI,IAAI,IAAI,IAAI,CAAC,CAAD,C;QACHb,IAAI,KAAK,CAAE,GAAE,C;QACb,IAAI,CAAC,QAAQ,CAAC,CAAD,CAAI,IAAG,CAAC,QAAQ,CAAC,EAAD,CAA7B,C;UAAmC,OAAO,IAAI,IAAI,CAAC,IAAI,IAAI,CAAC,CAAD,CAAI,GAAE,IAAI,IAAnB,C;QACID,OAAgB,CAAR,CAAE,GAAE,EAAl,IAAE,C;O;KAI1B,IAAI,OAAO,IAAI,KAAM,KAAl,WAAzB,C;MACI,IAAI,KAAM,GAAE,a;QACR,IAAI,IAAI,IAAI,CAAC,CAAD,CAAI,GAAE,cAAIB,C;UACI,IAAI,SAAS,C;UACb,IAAI,IAAI,IAAI,CAAC,CAAD,CAAI,GAAE,cAAIB,C;YACI,MAAO,IAAI,CAAE,GAAE,CAAE,GAAE,CAAG,GAAE,C;WAE5B,OAAO,M;;UAGP,IAAI,IAAI,IAAI,IAAI,CAAC,CAAC,CAAF,CAAhB,EAAsB,IAAI,IAAI,IAAI,CAAC,CAAC,CAAF,C;UACIC,OAAO,CAAE,KAAl,QAAS,GAAE,CAAF,GAAM,CAAE,KAAl,QAAS,GAAE,EA AF,GA Ae,CAAP,CAAE,GAAE,CAAG,KAAG,CAAE,GAAE,CAAP,C;;O;KAQtE,IAAI,OAAO,IAAI,MAAO,KAAl,WAA1B,C;MACI,IAAI,QAAQ,a;QACR,IAAI,CAAE,IAAG,CAAC,cAAV,C;UAEI,IAAI,CAAE,GAAE,oBAAR,C;YAEI,IAAI,CAAE,GAAE,oBAAR,C;cAGI,OAAO,IAAI,IAAI,CAAC,CAAD,CAAI,GAAE,IAAI,

I;;cAKzB,OAAO,IAAI,IAAI,CAAC,CAAE,GAAE,CAAE,GAAG,CAAE,IAAG,CAAE,GAAE,CAAP,CAAZ,C;;;  
YAKnB,OAAO,IAAI,IAAI,CAAC,CAAE,GAAE,IAAI,KAAK,CAAC,CAAE,GAAE,CAAE,GAAE,CAAT,CAAD  
,C;;eAGIB,IAAI,CAAE,IAAG,CAAC,cAAV,C;UAED,OAAO,CAAC,KAAK,CAAC,CAAC,CAAF,C;;UAKb,IAA  
I,SAAS,C;UACb,IAAI,IAAI,IAAI,CAAC,CAAD,CAAI,IAAG,cAAAnB,C;YAEI,IAAI,KAAK,CAAE,GAAE,CAA  
E,GAAE,C;YAEjB,MAAO,IAAG,EAAG,GAAE,C;WAEhB,OAAO,M;;O;MAGf,IAAI,MAAO,GAAE,K;KAEjB,I  
AAI,OAAO,IAAI,MAAO,KAAI,WAA1B,C;MACI,IAAI,MAAO,GAAE,a;QACT,IAAI,CAAE,GAAE,CAAR,C;U  
AEI,OAAO,G;eAEN,IAAI,CAAE,GAAE,CAAE,IAAG,cAAb,C;UAED,IAAI,CAAE,GAAE,oBAAR,C;YAGI,OA  
AO,IAAI,IAAI,CAAC,CAAD,CAAI,GAAE,IAAI,I;;YAlzB,OAAO,IAAI,IAAI,CAAC,CAAE,GAAE,IAAI,KAAK  
,CAAC,CAAE,GAAE,CAAE,GAAE,CAAT,CAAd,C;;;UAKnB,IAAI,IAAI,IAAI,KAAK,CAAC,CAAE,GAAE,C  
AAL,C;UAEjB,IAAI,SAAS,C;UACb,IAAI,CAAE,IAAG,cAAT,C;YAEI,IAAI,KAAK,CAAE,GAAE,CAAE,GAA  
E,C;YAEjB,MAAO,IAAG,EAAG,GAAE,E;WAGnB,OAAO,IAAI,KAAK,CAAC,CAAD,CAAI,GAAE,M;;O;KAI  
IC,IAAI,OAAO,IAAI,MAAO,KAAI,WAA1B,C;MACI,IAAI,MAAO,GAAE,a;QACT,IAAI,IAAI,IAAI,CAAC,CA  
AD,CAAI,GAAE,cAAIB,C;UACI,IAAI,SAAS,C;UACb,IAAI,IAAI,IAAI,CAAC,CAAD,CAAI,GAAE,cAAIB,C;Y  
ACI,MAAO,IAAI,CAAE,GAAE,CAAE,GAAE,CAAG,GAAE,C;WAE5B,OAAO,M;SAEX,OAAO,IAAI,IAAI,CA  
AS,CAAP,CAAE,GAAE,CAAG,KAAG,CAAE,GAAE,CAAP,CAAT,CAAoB,GAAE,C;O;KAG7C,IAAI,OAAO,I  
AAI,MAAO,KAAI,WAA1B,C;MACI,IAAI,MAAO,GAAE,a;QACT,IAAI,IAAI,IAAI,CAAC,CAAD,CAAI,GAAE  
,cAAIB,C;UACI,IAAI,KAAK,CAAE,GAAE,C;UACb,IAAI,KAAK,EAAG,GAAE,C;UACd,IAAI,KAAK,EAAG,  
GAAE,C;UAEd,OAAQ,CAAC,EAAG,GAAE,CAAE,GAAE,EAAG,GAAE,CAAE,GAAE,EAAG,GAAE,CAAE,G  
AAE,C;SAEXc,OAAO,IAAI,IAAI,CAAC,CAAE,GAAE,CAAL,C;O;KAGvB,IAAI,OAAO,IAAI,MAAO,KAAI,W  
AA1B,C;MACI,IAAI,MAAO,GAAE,a;QACT,IAAI,IAAI,IAAI,CAAC,CAAD,CAAI,GAAE,cAAIB,C;UACI,IAAI  
,KAAK,CAAE,GAAE,C;UACb,IAAI,KAAK,EAAG,GAAE,C;UACd,IAAI,KAAK,EAAG,GAAE,C;UAEd,OAAQ  
,EAAG,GAAE,EAAG,GAAE,EAAG,GAAE,CAAE,GAAE,EAAG,GAAE,CAAE,GAAE,C;SAEXc,OAAO,IAAI,I  
AAI,CAAC,CAAD,CAAI,GAAE,C;O;MAG/B,G;EACF,IAAI,OAAO,IAAI,MAAO,KAAI,WAA1B,C;IACI,IAAI,  
MAAO,GAAE,Y;MACT,IAAI,IAAI,C;MACR,IAAI,SAAS,SAAS,O;MAEtB,KAAK,IAAI,IAAI,CAAb,EAAGB,C  
AAE,GAAE,MAApB,EAA4B,CAAC,EAA7B,C;QACI,IAAI,SAAS,CAAC,CAAD,CAAI,KAAI,QAAS,IAAG,SA  
AS,CAAC,CAAD,CAAI,KAAI,CAAC,QAAnD,C;UACI,OAAO,Q;SAEX,CAAE,IAAG,SAAS,CAAC,CAAD,CA  
AI,GAAE,SAAS,CAAC,CAAD,C;;MAEjC,OAAO,IAAI,KAAK,CAAC,CAAD,C;K;GAGxB,IAAI,OAAO,IAAI,  
MAAO,KAAI,WAA1B,C;IACI,IAAI,MAAO,GAAE,a;MACT,OAAO,IAAI,IAAI,CAAC,CAAD,CAAI,GAAE,IA  
AI,O;K;GAGjC,IAAI,OAAO,IAAI,KAAM,KAAI,WAAzB,C;IACI,IAAI,KAAM,GAAE,a;MACR,OAAO,IAAI,IA  
AI,CAAC,CAAD,CAAI,GAAE,IAAI,M;K;GAGjC,IAAI,OAAO,IAAI,MAAO,KAAI,WAA1B,C;IACI,IAAI,MAA  
O,GAAG,oB;MACV,OAAO,a;QACH,IAAI,SAAS,CAAE,KAAI,C;QACnB,IAAI,MAAO,KAAI,CAAf,C;UACI,O  
AAO,E;SAEX,OAAO,EAAG,IAAG,GAAG,CAAC,MAAD,CAAS,GAAE,GAAL,GAAE,CAAvB,CAAoB,GAAE,  
C;O;KAE5C,CAAC,IAAI,IAAL,EAAW,IAAI,IAAf,C;GAIN,IAAI,OAAO,WAAW,OAAQ,KAAI,WAAIC,C;IACI  
,WAAW,OAAQ,GAAE,a;MACjB,OAAO,CAAE,IAAG,IAAK,IAAG,CAAC,UAAW,IAAG,IAAK,IAAG,CAAC,  
UAAU,UAAW,KAAI,SAAS,UAAU,U;K;GAlhG,IAAI,OAAO,KAAK,UAAU,KAAM,KAAI,WAApC,C;IAEI,M  
AAM,eAAe,CAAC,KAAK,UAAAN,EAakB,MAAlB,EAA0B,QACpC,iB;MAGH,IAAI,IAAK,IAAG,IAAZ,C;QAC  
I,MAAM,IAAI,SAAJ,CAAc,6BAAd,C;OAGV,IAAI,IAAI,MAAM,CAAC,IAAD,C;MAGd,IAAI,MAAM,CAAC,  
OAAQ,KAAI,C;MAGvB,IAAI,QAAQ,SAAS,CAAC,CAAD,C;MACrB,IAAI,gBAAgB,KAAM,IAAG,C;MAG7B,  
IAAI,IAAI,aAAc,GAAE,CAAE,GACIB,IAAI,IAAI,CAAC,GAAL,GAAE,aAAP,EAASB,CAAtB,CADU,GAEIB,I  
AAI,IAAI,CAAC,aAAD,EAAGB,GAAhB,C;MAGhB,IAAI,MAAM,SAAS,CAAC,CAAD,C;MACnB,IAAI,cAAc,  
GAAL,KAAI,SAAU,GACIB,GADkB,GACZ,GAAL,IAAG,C;MAG/B,IAAI,aAAa,WAAy,GAAE,CAAE,GACHB,I  
AAI,IAAI,CAAC,GAAL,GAAE,WAAp,EAaoB,CAApB,CADQ,GAehB,IAAI,IAAI,CAAC,WAAD,EAAC,GAAD,  
C;MAGzB,OAAO,CAAE,GAAE,UAAx,C;QACI,CAAC,CAAC,CAAD,CAAI,GAAE,K;QACP,CAAC,E;;MAIL,  
OAAO,C;KAvCgC,CAA1B,C;GA4HvB,CahFD,Y;IACG,yC;MACI,IAAI,MAAO,GAAE,CAAb,C;QAAgB,OAA  
O,IAAI,IAAI,CAAC,CAAD,EAAL,MAAO,GAAE,MAAb,C;MAC/B,OAAO,IAAI,IAAI,CAAC,MAAD,EAAS,M  
AAT,C;K;IAEnB,qC;MACI,IAAI,OAAO,GAAL,KAAI,WAAAnB,C;QACI,GAAL,GAAE,IAAI,O;OAEd,KAAM,GA  
AE,eAAe,CAAC,KAAM,IAAG,CAAV,EAAa,IAAI,OAAjB,C;MACvB,GAAL,GAAE,IAAI,IAAI,CAAC,KAAD,E  
AAQ,eAAe,CAAC,GAAD,EAAM,IAAI,OAAV,CAAvB,C;MACd,OAAO,IAAI,IAAI,YAAR,CAAqB,IAAI,SAAS

,CAAC,KAAD,EAAQ,GAAR,CAAIC,C;K;IAGX,IAAI,SAAS,CAAC,SAAD,EAAy,UAAZ,EAAwB,WAAxB,EA  
AqC,UAArC,EAAiD,YAAjD,EAA+D,YAA/D,C;IACb,KAAK,IAAI,IAAI,CAAb,EAAGB,CAAe,GAAE,MAAM,  
OAA1B,EAAmC,EAAE,CAArC,C;MACI,IAAI,aAAa,MAAM,CAAC,CAAD,C;MACvB,IAAI,OAAO,UAAU,UA  
AU,KAAM,KAAl,WAAzC,C;QACI,MAAM,eAAe,CAAC,UAAU,UAAx,EAAuB,MAAvB,EAA+B,QACzC,KA  
AK,UAAU,KAD0B,CAA/B,C;OAIzB,IAAI,OAAO,UAAU,UAAU,MAAO,KAAl,WAA1C,C;QACI,MAAM,eAA  
e,CAAC,UAAU,UAAx,EAAuB,OAAvB,EAAGC,QAC1C,eAD0C,CAAhC,C;;MAQJ,CAApB,Y;OAAc,MAAM,  
CAAC,IAAD,EAAO,IAAI,UAAJ,CAAe,CAAf,CAAP,E;;MAErB,IAAI,QAAQ,QAAQ,UAAU,M;MAC9B,MAA  
M,eAAe,CAAC,QAAQ,UAAU,EAAGB,OAARB,EAA8B,QACxC,uB;QACH,OAAO,KAAK,KAAK,CAAC,IAAD,  
EAAO,IAAP,EAAa,EAAE,MAAM,KAAK,CAAC,KAAD,CAA1B,C;OAF0B,CAA9B,C;;IASzB,KAAK,IAAI,IA  
AI,CAAb,EAAGB,CAAe,GAAE,MAAM,OAA1B,EAAmC,EAAE,CAArC,C;MACI,IAAI,aAAa,MAAM,CAAC,C  
AAD,C;MACvB,IAAI,OAAO,UAAU,UAAU,IAAK,KAAl,WAAxC,C;QACI,MAAM,eAAe,CAAC,UAAU,UAA  
X,EAAuB,MAAvB,EAA8B,QACxC,0B;UACH,OAAO,EAAE,MAAM,KAAK,CAAC,IAAD,CAAM,IAAI,CAAC,  
QAAD,EAAW,IAAX,C;SAFa,CAA9B,C;;IAU7B,IAAI,uBAAuB,gB;MACvB,IAAI,CAAe,GAAE,CAAR,C;QAA  
W,OAAO,E;MACIB,IAAI,CAAe,GAAE,CAAR,C;QAAW,OAAO,C;MAEIB,IAAI,CAAe,KAAl,CAAV,C;QACI,  
IAAI,CAAe,KAAl,CAAV,C;UAAa,OAAO,C;QAEpB,IAAI,KAAK,CAAe,GAAE,C;QACb,OAAO,EAAG,KAAl,  
CAAe,GAAE,CAAe,GAAE,CAAF,GAAO,EAAG,GAAE,CAAe,GAAE,EAAG,GAAO,C;OAG7C,OAAO,CAAe,  
KAAl,CAAe,GAAG,CAAe,KAAl,CAAe,GAAE,CAAF,GAAM,CAAjB,GAAsB,E;K;IAGzC,KAAK,IAAI,IAAI,  
CAAb,EAAGB,CAAe,GAAE,MAAM,OAA1B,EAAmC,EAAE,CAArC,C;MACI,IAAI,aAAa,MAAM,CAAC,CAA  
D,C;MACvB,IAAI,OAAO,UAAU,UAAU,KAAM,KAAl,WAAzC,C;QACI,MAAM,eAAe,CAAC,UAAU,UAAx,E  
AAuB,MAAvB,EAA+B,QACzC,2B;UACH,OAAO,KAAK,UAAU,KAAK,KAAK,CAAC,IAAD,EAAO,eAAGB,I  
AAG,oBAA1B,C;SAFY,CAA/B,C;;GAO/B,G;ECxXF,MAAM,KAAM,GAAE,QACH,OADG,aAEC,WAFD,UAG  
F,QAHE,C;EAMd,MAAM,WAAy,GAAE,2C;IACbB,IAAI,qBAAqB,MAAM,yBAAyB,CAAC,KAAD,EAAQ,YA  
AR,C;IACxD,IAAI,kBAAmB,IAAG,IAAK,IAAG,kBAaKb,IAAK,IAAG,IAA5D,C;MACI,OAAO,kBAaKb,IAAI,  
KAAK,CAAC,UAAD,C;KAGtC,kBAAmB,GAAE,MAAM,yBAAyB,CAAC,UAAD,EAAa,YAAb,C;IACpD,IAAI,  
kBAAmB,IAAG,IAAK,IAAG,OAAQ,IAAG,kBAA7C,C;MACI,OAAO,UAAU,CAAC,YAAD,C;KAGrB,OAAO,  
MAAM,WAAW,CAAC,UAAD,EAAa,MAAM,eAAe,CAAC,KAAD,CAAIC,EAA2C,YAA3C,C;G;EAG5B,MAA  
M,WAAy,GAAE,kD;IACbB,IAAI,qBAAqB,MAAM,yBAAyB,CAAC,KAAD,EAAQ,YAAR,C;IACxD,IAAI,kBA  
AmB,IAAG,IAAK,IAAG,kBAaKb,IAAK,IAAG,IAA5D,C;MACI,kBAaKb,IAAI,KAAK,CAAC,UAAD,EAAa,K  
AAb,C;MAC3B,M;KAGJ,kBAAmB,GAAE,MAAM,yBAAyB,CAAC,UAAD,EAAa,YAAb,C;IACpD,IAAI,kBAA  
mB,IAAG,IAAK,IAAG,OAAQ,IAAG,kBAA7C,C;MACI,UAAU,CAAC,YAAD,CAAe,GAAE,K;MAC3B,M;KA  
GJ,MAAM,WAAW,CAAC,UAAD,EAAa,MAAM,eAAe,CAAC,KAAD,CAAIC,EAA2C,YAA3C,EAAYD,KAAzD  
,C;G;EAGrB,iD;IACI,IAAI,IAAK,KAAl,KAAb,C;MAAoB,OAAO,I;IAE3B,IAAI,WAAW,IAAI,W;IACnB,IAAI,  
QAAS,IAAG,IAAhB,C;MACI,IAAI,aAAa,QAAQ,W;MACzB,KAAK,IAAI,IAAI,CAAb,EAAGB,CAAe,GAAE,U  
AAU,OAA9B,EAAuC,CAAC,EAAxC,C;QACI,IAAI,0BAA0B,CAAC,UAAU,CAAC,CAAD,CAAX,EAAGB,KA  
AhB,CAA9B,C;UACI,OAAO,I;;KAKnB,IAAI,iBAAiB,IAAI,UAAW,IAAG,IAAK,GAAE,MAAM,eAAe,CAAC,I  
AAI,UAAU,CAAvB,GAA0C,I;IACtF,IAAI,mBAAmB,cAAe,IAAG,IAAK,GAAE,cAAc,YAAhB,GAA+B,I;IAC7  
E,OAAO,gBAAiB,IAAG,IAAK,IAAG,0BAA0B,CAAC,gBAAD,EAAmB,KAAmB,C;G;EASjE,MAAM,OAAQ,G  
AAE,yB;IACZ,IAAI,KAAM,KAAl,MAAd,C;MACI,QAAQ,OAAO,MAAf,C;aACS,Q;aACA,Q;aACA,S;aACA,U;  
UACD,OAAO,I;gBAEP,OAAO,MAAO,YAAW,M;;KAIRc,IAAI,MAAO,IAAG,IAAK,IAAG,KAAM,IAAG,IAA  
K,KAAl,OAAO,MAAO,KAAl,QAAS,IAAG,OAAO,MAAO,KAAl,UAApD,CAApC,C;MACI,OAAO,K;KAGX,I  
AAI,OAAO,KAAM,KAAl,UAAW,IAAG,MAAO,YAAW,KAArD,C;MACI,OAAO,I;KAGX,IAAI,QAAQ,MAA  
M,eAAe,CAAC,KAAD,C;IACjC,IAAI,cAAc,KAAM,IAAG,IAAK,GAAE,KAAK,YAAP,GAAsB,I;IACtD,IAAI,  
WAAy,IAAG,IAAK,IAAG,YAAa,IAAG,WAA3C,C;MACI,IAAI,WAAW,WAAW,W;MAC1B,IAAI,QAAQ,KA  
AM,KAAl,MAAM,KAAK,OAAjC,C;QACI,OAAO,MAAO,KAAl,K;QAI1B,IAAI,gBAAgB,KAAK,W;IAGzB,IA  
AI,aAAc,IAAG,IAArB,C;MACI,OAAO,MAAO,YAAW,K;KAG7B,IAAI,aAAa,KAAM,KAAl,MAAM,KAAK,U  
AAW,IAAG,MAAM,YAAa,IAAG,IAA1E,C;MACI,OAAO,0BAA0B,CAAC,MAAM,YAAP,EAAGB,KAArB,C;K  
AGrC,OAAO,K;G;EAGX,MAAM,SAAU,GAAE,a;IACd,OAAO,OAAO,CAAe,IAAG,QAAS,IAAG,CAAe,YAA  
W,MAAM,K;G;EAGtD,MAAM,OAAQ,GAAE,iB;IACZ,OAAO,KAAM,YAAW,MAAM,U;G;EAGIC,MAAM,aA

Ac,GAAE,iB;IACIB,IAAI,OAAO,OAAO,K;IAEIB,OAAO,IAAK,KAAL,QAAS,IACIB,IAAK,KAAL,SAAU,IACn  
B,MAAM,SAAS,CAAC,KAAD,CAAQ,IACvB,MAAM,OAAO,CAAC,KAAD,EAAQ,MAAM,OAAO,WAArB,C;  
G;EAGxB,MAAM,eAAgB,GAAE,iB;IACpB,OAAO,OAAO,KAAM,KAAL,QAAS,IAAG,MAAM,OAAO,CAAC,  
KAAD,EAAQ,MAAM,OAAO,aArB,C;G;,,,,,;aCnDV,gB;,,,;ICrE3C,gB;MAkBI,4B;MAjBA,aAA6C,E;MAC7C,  
gBAAgD,C;K;4EAG5C,Y;MAAQ,iB;K;+EAGR,Y;MAAQ,oB;K;qCAEZ,iB;MAAyC,OAAQ,0BAAR,YAAQ,EA  
AU,KAAM,QAAb,C;K;4BAEjD,iB;MAAmC,gBAAS,K;K;8BAE5C,Y;MAA+B,OAAnc,MAAmC,kBAA8B,IA  
A9B,C;K;8BAE/B,Y;MAA0B,gB;K;IAE1B,0B;MAAA,8B;K;,,,;IAAA,sC;MAAA,qC;QAAA,oB;OAAA,8B;K;IDf  
J,mC;MAC4C,oBAAa,MAAS,IAAT,CAAb,EAA6B,SAA7B,C;K;gEAE5C,yB;MAAA,mB;MAAA,6B;QAC2D,Y  
AAa,QAAS,IAAT,C;QAIvD,Q;QAAA,OAAA,KAAM,OAAN,GAAa,CAAb,I;QAAb,aAAU,CAAV,iB;UACI,MA  
AM,CAAN,IALgF,IAKrE,CAAK,CAAL,C;QALwC,OAOhD,K;O;KARX,C;gEAGA,uB;MAEiB,Q;MAAA,OAA  
A,KAAM,OAAN,GAAa,CAAb,I;MAAb,aAAU,CAAV,iB;QACI,MAAM,CAAN,IAAW,KAAC,CAAL,C;MAEf,  
OAAO,K;K;IAGX,kC;MAliB,IAAN,I;MAFP,aAAsB,MAAe,IAAf,C;MACTB,gBAAkB,c;MAEd,IADS,IACT,mB  
ADS,IACT,EAAM,IAAN,E;QAAC,oBAAa,MAAb,EAAqB,KAARB,C;WACd,WAFS,IAET,S;QAAS,a;QAZA,U;  
QAAA,SAaqB,MAbf,OAAN,GAAa,CAAb,I;QAAb,aAAU,CAAV,mB;UAakC,MAZ9B,CAAM,CAAN,IAyS,C,IA  
Z3B,CAAK,CAAL,C;QAYH,OAAsB,M;MAHIC,W;K;2EAOJ,yB;MAAA,iC;MAAA,6B;QACoF,YAAa,aAAa,I  
AAb,EAAMB,KAAnB,C;QAlBhF,Q;QAAA,OAAA,KAAM,OAAN,GAAa,CAAb,I;QAAb,aAAU,CAAV,iB;UACI  
,MAAM,CAAN,IAiBoH,IAjBzG,CAAK,CAAL,C;QAIbIE,OafzE,K;O;KAcX,C;IAGA,+B;MAKiB,IAAN,I;MAF  
P,aAAa,IAAb,WAAa,CAAD,IAAC,C;MACb,gBAAkB,W;MAEd,IADS,IACT,mBADS,IACT,EAAM,IAAN,YAD  
S,IACT,EAAY,KAAZ,E;QAAqB,a;QA1BZ,U;QAAA,SA2BkB,MA3BZ,OAAN,GAAa,CAAb,I;QAAb,aAAU,CA  
AV,mB;UA2B+B,MA1B3B,CAAM,CAAN,IA0BmC,IA1BxB,CAAK,CAAL,C;QA0BH,OAAMB,M;MAF/B,W;  
K;qEAMJ,yB;MAAA,2B;MAAA,gC;MAAA,6B;QAGiB,Q;QADb,YAAY,UAAU,IAAV,EAAGB,IAAhB,C;QACC  
,OAAA,KAAM,OAAN,GAAa,CAAb,I;QAAb,aAAU,CAAV,iB;UACI,YACY,eAAK,CAAL,E;UACpB,KAAC,CA  
AC,CAAD,CAAG,GAAG,K;QAEP,OAAO,K;O;KARX,C;mFAWA,yB;MAAA,mB;MAAA,gC;MAAA,6B;QAGi  
B,Q;QADb,YAAY,QAAY,IAAZ,C;QACC,OAAA,KAAM,OAAN,GAAa,CAAb,I;QAAb,aAAU,CAAV,iB;UACI,  
YACY,eAAK,CAAL,E;UACpB,KAAC,CAAC,CAAD,CAAG,GAAG,K;QAEP,OAAO,K;O;KARX,C;IAWA,+B;  
MAliB,IAAN,I;MAFP,aAAsB,MAAY,IAAZ,C;MACTB,gBAAkB,W;MAEd,IADS,IACT,mBADS,IACT,EAAM,I  
AAN,E;QAAC,oBAAa,MAAb,K;WACd,WAFS,IAET,S;QAAS,a;QA3DA,U;QAAA,SA4DkB,MA5DZ,OAAN,G  
AAa,CAAb,I;QAAb,aAAU,CAAV,mB;UA4D+B,MA3D3B,CAAM,CAAN,IA2DmC,IA3DxB,CAAK,CAAL,C;Q  
A2DH,OAAMB,M;MAH/B,W;K;qEAOJ,yB;MAAA,2B;MAAA,6B;QAC2E,YAAa,UAAU,IAAV,EAAGB,KAAb  
B,C;QAJEvE,Q;QAAA,OAAA,KAAM,OAAN,GAAa,CAAb,I;QAAb,aAAU,CAAV,iB;UACI,MAAM,CAAN,IAg  
EwG,IAhE7F,CAAK,CAAL,C;QAgEwD,OA9DhE,K;O;KA6DX,C;IAGA,wC;MACiB,Q;MAAA,OAAA,KAAM,  
OAAN,GAAa,CAAb,I;MAAb,aAAU,CAAV,iB;QACI,MAAM,CAAN,IAAW,S;MAEf,OAAO,K;K;IEIFX,iC;MA  
AA,qC;MAEI,iBAC8B,Q;MAE9B,iBAC8B,sB;MAE9B,yBAEsC,MAAM,G;MAE5C,yBAEsC,CAAC,GAAD,GA  
AO,G;MAE7C,WAEwB,EAAE,MAAM,GAAR,C;MAExB,kBACuB,C;MAEvB,iBACsB,E;K;IAxB1B,6C;MAA  
A,4C;QAAA,2B;OAAA,qC;K;IA2BA,gC;MAAA,oC;MAEI,iBAC6B,O;MAE7B,iBAC6B,Y;MAE7B,yBAEqC,M  
AAO,G;MAE5C,yBAEqC,CAAC,GAAD,GAAQ,G;MAE7C,WAEuB,EAAE,MAAO,GAAT,C;MAEvB,kBACuB,  
C;MAEvB,iBACsB,E;K;IAxB1B,4C;MAAA,2C;QAAA,0B;OAAA,oC;K;IA2BA,8B;MAAA,kC;MAEI,iBACqB,  
W;MAErB,iBACqB,U;MAErB,kBACuB,C;MAEvB,iBACsB,E;K;IAZ1B,0C;MAAA,yC;QAAA,wB;OAAA,kC;  
K;IAeA,+B;MAAA,mC;MAEI,iBACJ,MAAM,KAAoB,U;MAEtB,iBACJ,MAAM,KAAoB,U;MAEtB,kBACuB,C;  
MAEvB,iBACsB,E;K;IAZ1B,2C;MAAA,0C;QAAA,yB;OAAA,mC;K;IAeA,gC;MAAA,oC;MAEI,iBACuB,U;M  
AEvB,iBACuB,K;MAEvB,kBACuB,C;MAEvB,iBACsB,E;K;IAZ1B,4C;MAAA,2C;QAAA,0B;OAAA,oC;K;IAe  
A,+B;MAAA,mC;MAEI,iBACsB,Q;MAEtB,iBACsB,G;MAEtB,kBACuB,C;MAEvB,iBACsB,C;K;IAZ1B,2C;M  
AAA,0C;QAAA,yB;OAAA,mC;K;IAeA,+B;MAAA,mC;MAEI,iBACmC,C;MAEnC,iBACmC,K;MAEnC,0BAC4  
C,K;MAE5C,0BAC4C,K;MAE5C,yBAC2C,K;MAE3C,yBAC2C,K;MAE3C,qBACuC,uB;MAEvC,qBACuC,sB;M  
AEvC,kBACuB,C;MAEvB,iBACsB,E;K;IA9B1B,2C;MAAA,0C;QAAA,yB;OAAA,mC;K;IAiCA,iC;MAAA,qC;  
K;IAAA,6C;MAAA,4C;QAAA,2B;OAAA,qC;K;IAEA,kC;MAAA,sC;K;IAAA,8C;MAAA,6C;QAAA,4B;OAA  
A,sC;K;,,,,,;aCkkuBoB,gB;,,,;cC/ntB0C,mB;gBAyEvC,yB;eAAyB,wB;uBAGbZB,gC;sBAA  
wB,+B;mCA4JjC,qB;mCA5ImC,qB;kBAQ1B,2B;iBAA0B,0B;,,,;eC3YgB,wB;sBCoBA,sB;iBCnBA,0B;,,,;aC5P8

B,e,,,,,,,,,,,,,gCCiDhD,yC;+BCVA,uC;+BCAA,sC;;gCCyJ/B,+B;+BAIW,sC;gCCqwCc,+B;0BAHvB,kC;uBAr6B  
O,gC;yBA8WD,iC;0BACA,mC;yBA4JA,iC;gCAmZP,oC;+BAbc,oC;+BAEC,+B;yBAEQ,kC;;gBCr0C6C,yB;,,,,;  
.....  
.....IC/ErF,kD;MAMuF,wC;K;IANvF,4CAOI,Y;MAAuC,8B;K;IAP3C,8E;ICGA,k  
D;MAQuF,wC;K;IARvF,4CASI,Y;MAAuC,8B;K;IAT3C,8E;0FbOA,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,  
qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAA  
I,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB  
;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,C  
AAJ,C;K;0FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;M  
AQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CA  
AJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MA  
QI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;0FAGX,qB;MAQI,OAAO,UAAI,CAAJ,  
C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,  
OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;  
K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,O  
AAO,UAAI,CAAJ,C;K;0FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;  
4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OA  
AO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4  
FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;0FAGX,qB;MAQI,OAA  
O,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4F  
AGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,  
UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAGX,qB;MAQI,OAAO,UAAI,CAAJ,C;K;4FAG  
X,qB;MAQI,OAAO,UAAI,CAAJ,C;K;IAGX,sC;MAII,OAAO,mBAAQ,OAAR,KAAoB,C;K;IAG/B,wC;MAII,OA  
AO,qBAAQ,OAAR,KAAoB,C;K;IAG/B,wC;MAII,OAAO,qBAAQ,OAAR,KAAoB,C;K;IAG/B,wC;MAII,OAAO,  
qBAAQ,OAAR,KAAoB,C;K;IAG/B,wC;MAII,OAAO,qBAAQ,OAAR,KAAoB,C;K;IAG/B,wC;MAOI,OAAO,qB  
AAQ,OAAR,KAAoB,C;K;IAG/B,wC;MAOI,OAAO,qBAAQ,OAAR,KAAoB,C;K;IAG/B,wC;MAII,OAAO,qB  
AAQ,OAAR,KAAoB,C;K;IAG/B,wC;MAII,OAAO,qBAAQ,OAAR,KAAoB,C;K;0GakE/B,yB;MAAA,8D;MAAA,  
iD;QAOI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAAsC,UAAI,KAAJ,CAATC,GAAAsD,aAAa,KAAb,C;O;K  
APjE,C;sGAUA,yB;MAAA,8D;MAAA,iD;QAOI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAAsC,UAAI,K  
AAJ,CAATC,GAAAsD,aAAa,KAAb,C;O;KAPjE,C;sGAUA,yB;MAAA,8D;MAAA,iD;QAOI,OAAW,SAAS,CAAT,  
IAAc,SAAS,wBAA3B,GAAAsC,UAAI,KAAJ,CAATC,GAAAsD,aAAa,KAAb,C;O;KAPjE,C;sGAUA,yB;MAAA,8D;  
MAAA,iD;QAOI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAAsC,UAAI,KAAJ,CAATC,GAAAsD,aAAa,KAA  
b,C;O;KAPjE,C;sGAUA,yB;MAAA,8D;MAAA,iD;QAOI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAAsC,U  
AAI,KAAJ,CAATC,GAAAsD,aAAa,KAAb,C;O;KAPjE,C;sGAUA,yB;MAAA,8D;MAAA,iD;QAOI,OAAW,SAAS,  
CAAT,IAAc,SAAS,wBAA3B,GAAAsC,UAAI,KAAJ,CAATC,GAAAsD,aAAa,KAAb,C;O;KAPjE,C;sGAUA,yB;MA  
AA,8D;MAAA,iD;QAOI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAAsC,UAAI,KAAJ,CAATC,GAAAsD,aA  
Aa,KAAb,C;O;KAPjE,C;sGAUA,yB;MAAA,8D;MAAA,iD;QAOI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,  
GAAAsC,UAAI,KAAJ,CAATC,GAAAsD,aAAa,KAAb,C;O;KAPjE,C;sGAUA,yB;MAAA,8D;MAAA,gC;MAAA,iD;  
QAOI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAAsC,UAAI,KAAJ,CAATC,GAAAsD,uBAAa,KAAb,E;O;K  
APjE,C;oGAUA,yB;MAAA,sD;MAAA,mC;QAOI,OAAY,UAAL,SAAK,EAAU,KAAV,C;O;KAPhB,C;qGAUA,y  
B;MAAA,qD;MAAA,mC;QAOI,OAAY,UAAL,SAAK,EAAU,KAAV,C;O;KAPhB,C;sGAUA,yB;MAAA,sD;MA  
AA,mC;QAOI,OAAY,UAAL,SAAK,EAAU,KAAV,C;O;KAPhB,C;sGAUA,yB;MAAA,sD;MAAA,mC;QAOI,OA  
AY,UAAL,SAAK,EAAU,KAAV,C;O;KAPhB,C;sGAUA,yB;MAAA,sD;MAAA,mC;QAOI,OAAY,UAAL,SAAK,  
EAAU,KAAV,C;O;KAPhB,C;sGAUA,yB;MAAA,sD;MAAA,mC;QAOI,OAAY,UAAL,SAAK,EAAU,KAAV,C;  
O;KAPhB,C;sGAUA,yB;MAAA,sD;MAAA,mC;QAOI,OAAY,UAAL,SAAK,EAAU,KAAV,C;O;KAPhB,C;sGA  
UA,yB;MAAA,sD;MAAA,mC;QAOI,OAAY,UAAL,SAAK,EAAU,KAAV,C;O;KAPhB,C;sGAUA,yB;MAAA,sD  
;MAAA,mC;QAOI,OAAY,UAAL,SAAK,EAAU,KAAV,C;O;KAPhB,C;8EAUA,gC;MAOW,sB;;QAYbS,Q;QAAb  
B,iD;UAAgB,cAAhB,e;UAAAsB,IAzbH,SAybO,CAAU,OAAY,CAAJ,C;YAAwB,qBAAO,O;YAAP,uB;;QAC9C,q

BAAO,I;;;MA1bP,yB;K;gFAGJ,gC;MAOW,sB;;QAubS,Q;QAAhB,iD;UAAgB,cAAhB,e;UAAsB,IAvbH,SAubO,CAAU,OAAV,CAAJ,C;YAAwB,qBAAO,O;YAAP,uB;;QAC9C,qBAAO,I;;;MAxbP,yB;K;gFAGJ,gC;MAOW,sB;;QAqbS,Q;QAAhB,iD;UAAgB,cAAhB,e;UAAsB,IArbH,SAqbO,CAAU,OAAV,CAAJ,C;YAAwB,qBAAO,O;YAAP,uB;;QAC9C,qBAAO,I;;;MATbP,yB;K;gFAGJ,gC;MAOW,sB;;QAmbS,Q;QAAhB,iD;UAAgB,cAAhB,e;UAAsB,IANbH,SAmbO,CAAU,OAAV,CAAJ,C;YAAwB,qBAAO,O;YAAP,uB;;QAC9C,qBAAO,I;;;MApbP,yB;K;gFAGJ,gC;MAOW,sB;;QAibS,Q;QAAhB,iD;UAAgB,cAAhB,e;UAAsB,IAjbH,SAibO,CAAU,OAAV,CAAJ,C;YAAwB,qBAAO,O;YAAP,uB;;QAC9C,qBAAO,I;;;MA1bP,yB;K;gFAGJ,gC;MAOW,sB;;QA+aS,Q;QAAhB,iD;UAAgB,cAAhB,e;UAAsB,IA/aH,SA+aO,CAAU,OAAV,CAAJ,C;YAAwB,qBAAO,O;YAAP,uB;;QAC9C,qBAAO,I;;;MAhbP,yB;K;gFAGJ,gC;MAOW,sB;;QA6aS,Q;QAAhB,iD;UAAgB,cAAhB,e;UAAsB,IA7aH,SA6aO,CAAU,OAAV,CAAJ,C;YAAwB,qBAAO,O;YAAP,uB;;QAC9C,qBAAO,I;;;MA9aP,yB;K;gFAGJ,gC;MAOW,sB;;QA2aS,Q;QAAhB,iD;UAAgB,cAAhB,e;UAAsB,IA3aH,SA2aO,CAAU,OAAV,CAAJ,C;YAAwB,qBAAO,O;YAAP,uB;;QAC9C,qBAAO,I;;;MA5aP,yB;K;gFAGJ,yB;MA4aA,oC;MAAA,gC;MA5aA,uC;QAOW,sB;;UayaS,Q;UAAhB,iD;YAAgB,cAAhB,OB;YAAsB,IAzaH,SAyaO,CAAU,oBAAV,CAAJ,C;cAAwB,qBAAO,O;cAAP,uB;;UAC9C,qBAAO,I;;;QA1aP,yB;O;KAPJ,C;sFAUA,yB;MAw1CA,0D;MAAA,+C;MAx1CA,uC;QAOW,qB;;UAu1CO,Q;UAAA,OA Aa,SAAR,sBAAQ,CAAb,W;UAAAd,OAAC,cAAd,C;YAAc,uB;YACV,cAAc,UAAK,KAAL,C;YACd,IAz1Cc,SAy1CV,CAAU,OAAV,CAAJ,C;cAAwB,oBAAO,O;cAAP,sB;;UAE5B,oBAAO,I;;;QA31CP,wB;O;KAPJ,C;wFAUA,yB;MA21CA,0D;MAAA,+C;MA31CA,uC;QAOW,qB;;UA01CO,Q;UAAA,OAAa,SAAR,sBAAQ,CAAb,W;UAAAd,OAAC,cAAd,C;YAAc,uB;YACV,cAAc,UAAK,KAAL,C;YACd,IA51Cc,SA41CV,CAAU,OAAV,CAAJ,C;cAAwB,oBAAO,O;cAAP,sB;;UAE5B,oBAAO,I;;;QA91CP,wB;O;KAPJ,C;wFAUA,yB;MA81CA,0D;MAAA,+C;MA91CA,uC;QAOW,qB;;UA61CO,Q;UAAA,OAAa,SAAR,sBAAQ,CAAb,W;UAAAd,OAAC,cAAd,C;YAAc,uB;YACV,cAAc,UAAK,KAAL,C;YACd,IA11Cc,SA+1CV,CAAU,OAAV,CAAJ,C;cAAwB,oBAAO,O;cAAP,sB;;UAE5B,oBAAO,I;;;QAj2CP,wB;O;KAPJ,C;wFAUA,yB;MAi2CA,0D;MAAA,+C;MAj2CA,uC;QAOW,qB;;UAag2CO,Q;UAAA,OAAa,SAAR,sBAAQ,CAAb,W;UAAAd,OAAC,cAAd,C;YAAc,uB;YACV,cAAc,UAAK,KAAL,C;YACd,IAI2Cc,SAk2CV,CAAU,OAAV,CAAJ,C;cAAwB,oBAAO,O;cAAP,sB;;UAE5B,oBAAO,I;;;QAp2CP,wB;O;KAPJ,C;wFAUA,yB;MAo2CA,0D;MAAA,+C;MAp2CA,uC;QAOW,qB;;UAm2CO,Q;UAAA,OAAa,SAAR,sBAAQ,CAAb,W;UAAAd,OAAC,cAAd,C;YAAc,uB;YACV,cAAc,UAAK,KAAL,C;YACd,IAr2Cc,SAq2CV,CAAU,OAAV,CAAJ,C;cAAwB,oBAAO,O;cAAP,sB;;UAE5B,oBAAO,I;;;QAv2CP,wB;O;KAPJ,C;wFAUA,yB;MAu2CA,0D;MAAA,+C;MAv2CA,uC;QAOW,qB;;UAs2CO,Q;UAAA,OAAa,SAAR,sBAAQ,CAAb,W;UAAAd,OAAC,cAAd,C;YAAc,uB;YACV,cAAc,UAAK,KAAL,C;YACd,IAx2Cc,SAw2CV,CAAU,OAAV,CAAJ,C;cAAwB,oBAAO,O;cAAP,sB;;UAE5B,oBAAO,I;;;QA12CP,wB;O;KAPJ,C;wFAUA,yB;MA02CA,0D;MAAA,+C;MA12CA,uC;QAOW,qB;;UAy2CO,Q;UAAA,OAAa,SAAR,sBAAQ,CAAb,W;UAAAd,OAAC,cAAd,C;YAAc,uB;YACV,cAAc,UAAK,KAAL,C;YACd,IA32Cc,SA22CV,CAAU,OAAV,CAAJ,C;cAAwB,oBAAO,O;cAAP,sB;;UAE5B,oBAAO,I;;;QA72CP,wB;O;KAPJ,C;wFAUA,yB;MA62CA,0D;MAAA,+C;MA72CA,uC;QAOW,qB;;UA42CO,Q;UAAA,OAAa,SAAR,sBAAQ,CAAb,W;UAAAd,OAAC,cAAd,C;YAAc,uB;YACV,cAAc,UAAK,KAAL,C;YACd,IA92Cc,SA82CV,CAAU,OAAV,CAAJ,C;cAAwB,oBAAO,O;cAAP,sB;;UAE5B,oBAAO,I;;;QAh3CP,wB;O;KAPJ,C;wFAUA,yB;MAg3CA,0D;MAAA,+C;MAAA,oC;MAh3CA,uC;QAOW,qB;;UA+2CO,Q;UAAA,OAAa,SAAR,sBAAQ,CAAb,W;UAAAd,OAAC,cAAd,C;YAAc,uB;YACV,cAAc,UAAK,KAAL,C;YACd,IAj3Cc,SAi3CV,CAAU,oBAAV,CAAJ,C;cAAwB,oBAAO,O;cAAP,sB;;UAE5B,oBAAO,I;;;QAn3CP,wB;O;KAPJ,C;IAUA,0B;MAKI,IA4uNO,qBAAQ,CA5uNf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAK,CAAL,C;K;IAGX,4B;MAKI,IA0uNO,qBAAQ,CA1uNf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAK,CAAL,C;K;IAGX,4B;MAKI,IAwuNO,qBAAQ,CAxuNf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAK,CAAL,C;K;IAGX,4B;MAKI,IASuNO,qBAAQ,CAtuNf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAK,CAAL,C;K;IAGX,4B;MAKI,IAouNO,qBAAQ,CApuNf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAK,CAAL,C;K;IAGX,4B;MAKI,IAkuNO,qBAAQ,CAluNf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAK,CAAL,C;K;IAGX,4B;MAKI,IAguNO,qBAAQ,CAhuNf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAK,CAAL,C;K;IAGX,4B;MAKI,IA8tNO,qBAAQ,CA9tNf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAK,CAAL,C;K;IAGX,4B;MAKI,IA4tNO,qBAAQ,CA5tNf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAK,CAAL,C;K;kFAGX,yB;MAAA,iE;MAAA,uC;QAKoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UAAsB,IAAI,

UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QACrD,MAAM,gCAAuB,mDAAvB,C;O;KANV,C;kFASA,yB;MAA  
A,iE;MAAA,uC;QAKoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UAAsB,IAAI,UAAU,OAA  
V,CAAJ,C;YAAwB,OAAO,O;;QACrD,MAAM,gCAAuB,mDAAvB,C;O;KANV,C;mFASA,yB;MAAA,iE;MAAA  
,uC;QAKoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UAAsB,IAAI,UAAU,OAAV,CAAJ,C;Y  
AAwB,OAAO,O;;QACrD,MAAM,gCAAuB,mDAAvB,C;O;KANV,C;mFASA,yB;MAAA,iE;MAAA,uC;QAKoB,  
Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UAAsB,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAA  
O,O;;QACrD,MAAM,gCAAuB,mDAAvB,C;O;KANV,C;mFASA,yB;MAAA,iE;MAAA,uC;QAKoB,Q;QAAhB,w  
BAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UAAsB,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QACrD  
,MAAM,gCAAuB,mDAAvB,C;O;KANV,C;mFASA,yB;MAAA,iE;MAAA,uC;QAKoB,Q;QAAhB,wBAAgB,SA  
AhB,gB;UAAgB,cAAA,SAAhB,M;UAAsB,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QACrD,MAAM,gC  
AAuB,mDAAvB,C;O;KANV,C;mFASA,yB;MAAA,iE;MAAA,uC;QAKoB,Q;QAAhB,wBAAgB,SAAhB,gB;UA  
AgB,cAAA,SAAhB,M;UAAsB,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QACrD,MAAM,gCAAuB,mDA  
AvB,C;O;KANV,C;mFASA,yB;MAAA,iE;MAAA,uC;QAKoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,S  
AAhB,M;UAAsB,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QACrD,MAAM,gCAAuB,mDAAvB,C;O;KA  
NV,C;mFASA,yB;MAAA,oC;MAAA,gC;MAAA,iE;MAAA,uC;QAKoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAg  
B,cAAhB,UAAgB,SAAhB,O;UAAsB,IAAI,UAAU,oBAAV,CAAJ,C;YAAwB,OAAO,O;;QACrD,MAAM,gCAAu  
B,mDAAvB,C;O;KANV,C;kGASA,yB;MAAA,iE;MAAA,uC;QASW,Q;QAAA,+B;;UAYS,U;UAAhB,uD;YAAg  
B,cAAhB,iB;YACI,aAbwB,SAaX,CAAU,OAAV,C;YACb,IAAI,cAAJ,C;cACI,8BAAO,M;cAAP,gC;;UAGR,8BA  
AO,I;;QAIBA,kC;QAAA,iB;UAAmC,MAAM,gCAAuB,8DAAvB,C;SAAhD,OAAO,I;O;KATX,C;8GAYA,gC;M  
ASoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,aAAa,UAAU,OAAV,C;QACb,IAAI,cA  
AJ,C;UACI,OAAO,M;;MAGf,OAAO,I;K;IAGX,gC;MAII,OAoiNO,qBAAQ,CApiNR,GAAe,IAAf,GAAyB,UAA  
K,CAAL,C;K;IAGpC,kC;MAII,OAqiNO,qBAAQ,CAriNR,GAAe,IAAf,GAAyB,UAAK,CAAL,C;K;IAGpC,kC;M  
AII,OAsiNO,qBAAQ,CAtiNR,GAAe,IAAf,GAAyB,UAAK,CAAL,C;K;IAGpC,kC;MAII,OAuiNO,qBAAQ,CAvi  
NR,GAAe,IAAf,GAAyB,UAAK,CAAL,C;K;IAGpC,kC;MAII,OAwiNO,qBAAQ,CAxiNR,GAAe,IAAf,GAAyB,U  
AAK,CAAL,C;K;IAGpC,kC;MAII,OAYiNO,qBAAQ,CAziNR,GAAe,IAAf,GAAyB,UAAK,CAAL,C;K;IAGpC,k  
C;MAII,OA0iNO,qBAAQ,CA1iNR,GAAe,IAAf,GAAyB,UAAK,CAAL,C;K;IAGpC,kC;MAII,OA2iNO,qBAAQ,  
CA3iNR,GAAe,IAAf,GAAyB,UAAK,CAAL,C;K;IAGpC,kC;MAII,OA4iNO,qBAAQ,CA5iNR,GAAe,IAAf,GAA  
yB,UAAK,CAAL,C;K;8FAGpC,gC;MAIoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,  
IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,O;;MACrD,OAAO,I;K;8FAGX,gC;MAIoB,Q;MAAhB,wBAAgB,S  
AAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,O;;MACrD,OAAO,I;  
K;+FAGX,gC;MAIoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,  
CAAJ,C;UAAwB,OAAO,O;;MACrD,OAAO,I;K;+FAGX,gC;MAIoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAgB,c  
AAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,O;;MACrD,OAAO,I;K;+FAGX,gC;MAIo  
B,Q;MAAhB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,O  
AAO,O;;MACrD,OAAO,I;K;+FAGX,gC;MAIoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QA  
AsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,O;;MACrD,OAAO,I;K;+FAGX,gC;MAIoB,Q;MAAhB,wBA  
AgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,O;;MACrD,O  
AAO,I;K;+FAGX,gC;MAIoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,  
OAAV,CAAJ,C;UAAwB,OAAO,O;;MACrD,OAAO,I;K;+FAGX,yB;MAAA,oC;MAAA,gC;MAAA,uC;QAIoB,Q  
;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UAAsB,IAAI,UAAU,oBAAV,CAAJ,C;YAA  
wB,OAAO,O;;QACrD,OAAO,I;O;KALX,C;wFAQA,yB;MAAA,8D;MAAA,iD;QAKI,OAAW,SAAS,CAAT,IAA  
c,SAAS,wBAA3B,GAAc,UAAI,KAJ,CAAtC,GAAcD,aAAa,KAAb,C;O;KALjE,C;0FAQA,yB;MAAA,8D;MA  
AA,iD;QAKI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAc,UAAI,KAJ,CAAtC,GAAcD,aAAa,KAAb,C;  
O;KALjE,C;0FAQA,yB;MAAA,8D;MAAA,iD;QAKI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAc,UAA  
I,KAJ,CAAtC,GAAcD,aAAa,KAAb,C;O;KALjE,C;0FAQA,yB;MAAA,8D;MAAA,iD;QAKI,OAAW,SAAS,CA  
AT,IAAc,SAAS,wBAA3B,GAAc,UAAI,KAJ,CAAtC,GAAcD,aAAa,KAAb,C;O;KALjE,C;0FAQA,yB;MAAA,  
8D;MAAA,iD;QAKI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAc,UAAI,KAJ,CAAtC,GAAcD,aAAa,K  
AAb,C;O;KALjE,C;0FAQA,yB;MAAA,8D;MAAA,iD;QAKI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAc





c,cAAAd,C;QAAC,uB;QACV,IAAI,YAAW,UAAK,KAAL,CAAF,C;UACI,OAAO,K;;MAGf,OAAO,E;K;IAGX,2C; MAIkB,Q;MAAA,OAAQ,WAAR,wBAAQ,CAAR,W;MAAd,OAAc,cAAAd,C;QAAC,uB;QACV,IAAI,YAAW,UA AK,KAAL,CAAF,C;UACI,OAAO,K;;MAGf,OAAO,E;K;IAGX,+B;MAMI,OA8jLO,qBAAQ,CA9jLR,GAAe,IAA f,GAAyB,UAAK,mBAAO,CAAP,IAAL,C;K;IAGpC,iC;MAMI,OA6jLO,qBAAQ,CA7jLR,GAAe,IAAf,GAAyB,U AAK,mBAAO,CAAP,IAAL,C;K;IAGpC,iC;MAMI,OA4jLO,qBAAQ,CA5jLR,GAAe,IAAf,GAAyB,UAAK,mBA AO,CAAP,IAAL,C;K;IAGpC,iC;MAMI,OA2jLO,qBAAQ,CA3jLR,GAAe,IAAf,GAAyB,UAAK,mBAAO,CAAP, IAAL,C;K;IAGpC,iC;MAMI,OA0jLO,qBAAQ,CA1jLR,GAAe,IAAf,GAAyB,UAAK,mBAAO,CAAP,IAAL,C;K; IAGpC,iC;MAMI,OAyjLO,qBAAQ,CAzjLR,GAAe,IAAf,GAAyB,UAAK,mBAAO,CAAP,IAAL,C;K;IAGpC,iC; MAMI,OAwjLO,qBAAQ,CAxjLR,GAAe,IAAf,GAAyB,UAAK,mBAAO,CAAP,IAAL,C;K;IAGpC,iC;MAMI,OA ujLO,qBAAQ,CAvjLR,GAAe,IAAf,GAAyB,UAAK,mBAAO,CAAP,IAAL,C;K;IAGpC,iC;MAMI,OAsjLO,qBAA Q,CAtjLR,GAAe,IAAf,GAAyB,UAAK,mBAAO,CAAP,IAAL,C;K;4FAGpC,yB;MAAA,0D;MAAA,+C;MAAA,u C;QAMkB,Q;QAAA,OAAa,SAAR,YAAL,SAAK,CAAQ,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAA c,UAAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,OAAO,I;O;KAVX,C;4FAaA, yB;MAAA,0D;MAAA,+C;MAAA,uC;QAMkB,Q;QAAA,OAAa,SAAR,YAAL,SAAK,CAAQ,CAAb,W;QAAd,O AAc,cAAAd,C;UAAc,uB;UACV,cAAc,UAAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;; QAEEnC,OAAO,I;O;KAVX,C;6FAaA,yB;MAAA,0D;MAAA,+C;MAAA,uC;QAMkB,Q;QAAA,OAAa,SAAR,YA AL,SAAK,CAAQ,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAAc,UAAK,KAAL,C;UACd,IAAI,UAAU, OAAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,OAAO,I;O;KAVX,C;6FAaA,yB;MAAA,0D;MAAA,+C;MAAA,uC; QAMkB,Q;QAAA,OAAa,SAAR,YAAL,SAAK,CAAQ,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAAc, UAAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,OAAO,I;O;KAVX,C;6FAaA,y B;MAAA,0D;MAAA,+C;MAAA,uC;QAMkB,Q;QAAA,OAAa,SAAR,YAAL,SAAK,CAAQ,CAAb,W;QAAd,OA Ac,cAAAd,C;UAAc,uB;UACV,cAAc,UAAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;Q AEEnC,OAAO,I;O;KAVX,C;6FAaA,yB;MAAA,0D;MAAA,+C;MAAA,uC;QAMkB,Q;QAAA,OAAa,SAAR,YAA L,SAAK,CAAQ,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAAc,UAAK,KAAL,C;UACd,IAAI,UAAU,O AAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,OAAO,I;O;KAVX,C;6FAaA,yB;MAAA,0D;MAAA,+C;MAAA,uC;Q AMkB,Q;QAAA,OAAa,SAAR,YAAL,SAAK,CAAQ,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAAc,U AAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,OAAO,I;O;KAVX,C;6FAaA,yB; MAAA,0D;MAAA,+C;MAAA,uC;QAMkB,Q;QAAA,OAAa,SAAR,YAAL,SAAK,CAAQ,CAAb,W;QAAd,OAAc ,cAAAd,C;UAAc,uB;UACV,cAAc,UAAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QAE nC,OAAO,I;O;KAVX,C;6FAaA,yB;MAAA,0D;MAAA,+C;MAAA,oC;MAAA,uC;QAMkB,Q;QAAA,OAAa,SAAR,YAAL,SAAK,CAAQ,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAAc,UAAK,KAAL,C;UACd,IAAI,U AAU,oBAAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,OAAO,I;O;KAVX,C;kFAaA,yB;MAAA,mC;MAAA,gD;MA AA,4B;QAQI,OAAO,kBAAO,cAAP,C;O;KARX,C;oFAWA,yB;MAAA,mC;MAAA,gD;MAAA,4B;QAQI,OAAO ,kBAAO,cAAP,C;O;KARX,C;oFAWA,yB;MAAA,mC;MAAA,gD;MAAA,4B;QAQI,OAAO,kBAAO,cAAP,C;O; KARX,C;oFAWA,yB;MAAA,mC;MAAA,gD;MAAA,4B;QAQI,OAAO,kBAAO,cAAP,C;O;KARX,C;oFAWA,yB; MAAA,mC;MAAA,gD;MAAA,4B;QAQI,OAAO,kBAAO,cAAP,C;O;KARX,C;oFAWA,yB;MAAA,mC;MAAA,gD; MAAA,4B;QAQI,OAAO,kBAAO,cAAP,C;O;KARX,C;oFAWA,yB;MAAA,mC;MAAA,gD;MAAA,4B;QAQI ,OAAO,kBAAO,cAAP,C;O;KARX,C;oFAWA,yB;MAAA,mC;MAAA,gD;MAAA,4B;QAQI,OAAO,kBAAO,cAA P,C;O;KARX,C;oFAWA,yB;MAAA,mC;MAAA,gD;MAAA,4B;QAQI,OAAO,kBAAO,cAAP,C;O;KARX,C;IAW A,qC;MAOI,IAoxKO,qBAAQ,CAPxKf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAI,MAAO,iBA AQ,gBAAR,CAAX,C;K;IAGX,qC;MAOI,IAgxKO,qBAAQ,CAhxKf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MAC V,OAAO,UAAI,MAAO,iBAAQ,gBAAR,CAAX,C;K;IAGX,sC;MAOI,IA4wKO,qBAAQ,CA5wKf,C;QACI,MAA M,2BAAuB,iBAAvB,C;MACV,OAAO,UAAI,MAAO,iBAAQ,gBAAR,CAAX,C;K;IAGX,sC;MAOI,IAwwKO,qB AAQ,CAxwKf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAI,MAAO,iBAAQ,gBAAR,CAAX,C;K; IAGX,sC;MAOI,IAowKO,qBAAQ,CAPwKf,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAI,MAAO ,iBAAQ,gBAAR,CAAX,C;K;IAGX,sC;MAOI,IAgwKO,qBAAQ,CAhwKf,C;QACI,MAAM,2BAAuB,iBAAvB,C; MACV,OAAO,UAAI,MAAO,iBAAQ,gBAAR,CAAX,C;K;IAGX,sC;MAOI,IA4vKO,qBAAQ,CA5vKf,C;QACI, MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,UAAI,MAAO,iBAAQ,gBAAR,CAAX,C;K;IAGX,sC;MAOI,IAwvK





AhB,UAAgB,SAAhB,O;UACI,IAAI,UAAU,oBAAV,CAAJ,C;YACI,IAAI,KAAJ,C;cAAW,OAAO,I;YACIB,SAA S,O;YACT,QAAQ,I;;QAGhB,IAAI,CAAC,KAAL,C;UAAy,OAAO,I;QACnB,OAAO,M;O;KADx,C;IAiBA,4B;M cvqGI,IAAI,Ed+qGI,KAAK,Cc/qGT,CAAJ,C;QACI,cd8qGc,sD;Qc7qGd,MAAM,gCAAYB,OAAQ,WAAjC,C;Od 8qGV,OAAO,oBAAoB,gBAAV,mBAAO,CAAP,IAAU,EAAC,CAAd,CAApB,C;K;IAGX,8B;McnrGI,IAAI,Ed2rG I,KAAK,Cc3rGT,CAAJ,C;QACI,cd0rGc,sD;QczrGd,MAAM,gCAAYB,OAAQ,WAAjC,C;Od0rGV,OAAO,sBAAo B,gBAAV,mBAAO,CAAP,IAAU,EAAC,CAAd,CAApB,C;K;IAGX,8B;Mc/rGI,IAAI,EdusGI,KAAK,CcvsGT,CA AJ,C;QACI,cdssGc,sD;QcrsGd,MAAM,gCAAYB,OAAQ,WAAjC,C;OdssGV,OAAO,sBAAoB,gBAAV,mBAAO, CAAP,IAAU,EAAC,CAAd,CAApB,C;K;IAGX,8B;Mc3sGI,IAAI,EdmtGI,KAAK,CcntGT,CAAJ,C;QACI,cdktGc, sD;QcjtGd,MAAM,gCAAYB,OAAQ,WAAjC,C;OdktGV,OAAO,sBAAoB,gBAAV,mBAAO,CAAP,IAAU,EAAC, CAAd,CAApB,C;K;IAGX,8B;McvtGI,IAAI,Ed+tGI,KAAK,Cc/tGT,CAAJ,C;QACI,cd8tGc,sD;Qc7tGd,MAAM,g CAAYB,OAAQ,WAAjC,C;Od8tGV,OAAO,sBAAoB,gBAAV,mBAAO,CAAP,IAAU,EAAC,CAAd,CAApB,C;K;I AGX,8B;McnuGI,IAAI,Ed2uGI,KAAK,Cc3uGT,CAAJ,C;QACI,cd0uGc,sD;QczuGd,MAAM,gCAAYB,OAAQ,W AAjC,C;Od0uGV,OAAO,sBAAoB,gBAAV,mBAAO,CAAP,IAAU,EAAC,CAAd,CAApB,C;K;IAGX,8B;Mc/uGI,I AAI,EduvGI,KAAK,CcuvGT,CAAJ,C;QACI,cdsvGc,sD;QcervGd,MAAM,gCAAYB,OAAQ,WAAjC,C;OdsVGV,O AAO,sBAAoB,gBAAV,mBAAO,CAAP,IAAU,EAAC,CAAd,CAApB,C;K;IAGX,8B;Mc3vGI,IAAI,EdmwGI,KAA K,CcnwGT,CAAJ,C;QACI,cdkwGc,sD;QcjuGd,MAAM,gCAAYB,OAAQ,WAAjC,C;OdkwGV,OAAO,sBAAoB,g BAAV,mBAAO,CAAP,IAAU,EAAC,CAAd,CAApB,C;K;IAGX,8B;McvwGI,IAAI,Ed+wGI,KAAK,Cc/wGT,CAA J,C;QACI,cd8wGc,sD;Qc7wGd,MAAM,gCAAYB,OAAQ,WAAjC,C;Od8wGV,OAAO,sBAAoB,gBAAV,mBAAO ,CAAP,IAAU,EAAC,CAAd,CAApB,C;K;IAGX,gC;McnxGI,IAAI,Ed2xGI,KAAK,Cc3xGT,CAAJ,C;QACI,cd0xG c,sD;QczxGd,MAAM,gCAAYB,OAAQ,WAAjC,C;Od0xGV,OAAO,gBAAgB,gBAAV,mBAAO,CAAP,IAAU,EA Ac,CAAd,CAAhB,C;K;IAGX,kC;Mc/xGI,IAAI,EduyGI,KAAK,CcuyGT,CAAJ,C;QACI,cdsyGc,sD;QcryGd,MAA M,gCAAYB,OAAQ,WAAjC,C;OdsyGV,OAAO,kBAAgB,gBAAV,mBAAO,CAAP,IAAU,EAAC,CAAd,CAAhB,C ;K;IAGX,kC;Mc3yGI,IAAI,EdmzGI,KAAK,CcnzGT,CAAJ,C;QACI,cdkzGc,sD;QcizGd,MAAM,gCAAYB,OAAQ ,WAAjC,C;OdkzGV,OAAO,kBAAgB,gBAAV,mBAAO,CAAP,IAAU,EAAC,CAAd,CAAhB,C;K;IAGX,kC;Mcvz GI,IAAI,Ed+zGI,KAAK,Cc/zGT,CAAJ,C;QACI,cd8zGc,sD;Qc7zGd,MAAM,gCAAYB,OAAQ,WAAjC,C;Od8zG V,OAAO,kBAAgB,gBAAV,mBAAO,CAAP,IAAU,EAAC,CAAd,CAAhB,C;K;IAGX,kC;Mcn0GI,IAAI,Ed20GI,K AAK,Cc30GT,CAAJ,C;QACI,cd00Gc,sD;Qcz0Gd,MAAM,gCAAYB,OAAQ,WAAjC,C;Od00GV,OAAO,kBAAg B,gBAAV,mBAAO,CAAP,IAAU,EAAC,CAAd,CAAhB,C;K;IAGX,kC;Mc/0GI,IAAI,Ed1GI,KAAK,Ccv1GT,CA AJ,C;QACI,cds1Gc,sD;Qcr1Gd,MAAM,gCAAYB,OAAQ,WAAjC,C;Ods1GV,OAAO,kBAAgB,gBAAV,mBAAO, CAAP,IAAU,EAAC,CAAd,CAAhB,C;K;IAGX,kC;Mc31GI,IAAI,Edm2GI,KAAK,Ccn2GT,CAAJ,C;QACI,cdk2G c,sD;Qcj2Gd,MAAM,gCAAYB,OAAQ,WAAjC,C;Odk2GV,OAAO,kBAAgB,gBAAV,mBAAO,CAAP,IAAU,EA Ac,CAAd,CAAhB,C;K;IAGX,kC;Mcv2GI,IAAI,Ed+2GI,KAAK,Cc/2GT,CAAJ,C;QACI,cd82Gc,sD;Qc72Gd,MA AM,gCAAYB,OAAQ,WAAjC,C;Od82GV,OAAO,kBAAgB,gBAAV,mBAAO,CAAP,IAAU,EAAC,CAAd,CAAhB ,C;K;IAGX,kC;Mcn3GI,IAAI,Ed23GI,KAAK,Cc33GT,CAAJ,C;QACI,cd03Gc,sD;Qcz3Gd,MAAM,gCAAYB,OA AQ,WAAjC,C;Od03GV,OAAO,kBAAgB,gBAAV,mBAAO,CAAP,IAAU,EAAC,CAAd,CAAhB,C;K;gGAGX,yB; MAAA,8D;MAAA,4C;MAAA,qD;MAAA,uC;QAMI,iBAAc,wBAAd,WAA+B,CAA/B,U;UACI,IAAI,CAAC,UA AU,UAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL,C;;QAGf,OAAO,W;O;KAXX,C; kGAcA,yB;MAAA,8D;MAAA,4C;MAAA,qD;MAAA,uC;QAMI,iBAAc,wBAAd,WAA+B,CAA/B,U;UACI,IAAI, CAAC,IAAU,UAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL,C;;QAGf,OAAO,W;O ;KAXX,C;kGAcA,yB;MAAA,8D;MAAA,4C;MAAA,qD;MAAA,uC;QAMI,iBAAc,wBAAd,WAA+B,CAA/B,U;U ACI,IAAI,CAAC,IAAU,UAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL,C;;QAGf,OA AO,W;O;KAXX,C;kGAcA,yB;MAAA,8D;MAAA,4C;MAAA,qD;MAAA,uC;QAMI,iBAAc,wBAAd,WAA+B,CA A/B,U;UACI,IAAI,CAAC,IAAU,UAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL, C;;QAGf,OAAO,W;O;KAXX,C;kGAcA,yB;MAAA,8D;MAAA,4C;MAAA,qD;MAAA,uC;QAMI,iBA Ac,wBAAd,WAA+B,CAA/B,U;UACI,IAAI,CAAC,IAAU,UAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CA AR,IAAL,C;;QAGf,OAAO,W;O;KAXX,C;kGAcA,yB;MAAA,8D;MAAA,4C;MAAA,qD;MAAA,uC;QAMI,iBA Ac,wBAAd,WAA+B,CAA/B,U;UACI,IAAI,CAAC,IAAU,UAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL,C;;QAGf,OAAO,W;O;KAXX,C;kGAcA,yB;MAAA,8D;MAAA,4C;MAAA,qD;MAAA,uC;



W;O;KAlhBX,C;gGASA,yB;MAAA,+D;MAAA,uC;QAQW,kBAAGB,gB;QAsGTV,gB;QADb,YAAY,C;QACZ,iD ;UAAa,WAAb,e;UA16SI,IAPGmC,SAoG/B,Eak6SkB,cAl6SIB,Eak6SkB,sBA16SIB,Wak6S2B,IA16S3B,CAAJ,C; YAA2C,sBAk6SZ,IA16SY,C;;QApg/C,OAsGO,W;O;KA9GX,C;kGAWA,yB;MAAA,+D;MAAA,uC;QAQW,kBA AgB,gB;QAqgTV,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,e;UA95SI,IAvGsC,SAuG1C,EA85SkB,cA95SIB,E A85SkB,sBA95SIB,WA85S2B,IA95S3B,CAAJ,C;YAA2C,sBA85SZ,IA95SY,C;;QAvG/C,OAYGO,W;O;KAjHX, C;kGAWA,yB;MAAA,+D;MAAA,uC;QAQW,kBAAGB,gB;QAogTV,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAA b,e;UA15SI,IA1GuC,SA0GnC,EA05SkB,cA15SIB,EA05SkB,sBA15SIB,WA05S2B,IA15S3B,CAAJ,C;YAA2C,sB A05SZ,IA15SY,C;;QA1G/C,OA4GO,W;O;KAphX,C;kGAWA,yB;MAAA,+D;MAAA,uC;QAQW,kBAAGB,gB;Q AmgTV,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,e;UA5SI,IA7GqC,SA6GjC,EAs5SkB,cAt5SIB,EAs5SkB,s BA5SIB,WAs5S2B,IAt5S3B,CAAJ,C;YAA2C,sBA5SZ,IAt5SY,C;;QA7G/C,OA+GO,W;O;KAvHX,C;kGAWA, yB;MAAA,+D;MAAA,uC;QAQW,kBAAGB,gB;QAkgTV,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,e;UA15SI, IAhHsC,SAGHIC,Eak5SkB,cAl5SIB,Eak5SkB,sBA15SIB,Wak5S2B,IA15S3B,CAAJ,C;YAA2C,sBAk5SZ,IA15S Y,C;;QAhh/C,OAKHO,W;O;KA1HX,C;kGAWA,yB;MAAA,+D;MAAA,uC;QAQW,kBAAGB,gB;QAigTV,gB;Q ADb,YAAY,C;QACZ,iD;UAAa,WAAb,e;UA94SI,IANuHc,SAmHnC,EA84SkB,cA94SIB,EA84SkB,sBA94SIB,W A84S2B,IA94S3B,CAAJ,C;YAA2C,sBA84SZ,IA94SY,C;;QAnH/C,OAqHO,W;O;KA7HX,C;kGAWA,yB;MAAA ,+D;MAAA,uC;QAQW,kBAAGB,gB;QAkgTV,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,e;UA14SI,IAtHwC,S AsHpC,EA04SkB,cA14SIB,EA04SkB,sBA14SIB,WA04S2B,IA14S3B,CAAJ,C;YAA2C,sBA04SZ,IA14SY,C;;QA tH/C,OAwhO,W;O;KAhIX,C;kGAWA,yB;MAAA,+D;MAAA,uC;QAQW,kBAAGB,gB;QA+/SV,gB;QADb,YAA Y,C;QACZ,iD;UAAa,WAAb,e;UA4SI,IAzHyC,SAyHrC,EAs4SkB,cAt4SIB,EAs4SkB,sBA4SIB,WAs4S2B,IAt4 S3B,CAAJ,C;YAA2C,sBA4SZ,IAt4SY,C;;QAZH/C,OA2HO,W;O;KANIX,C;kGAWA,yB;MAAA,+D;MA2HA,g C;MAo4SA,oC;MA//SA,uC;QAQW,kBAAGB,gB;QA8/SV,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,0B;UAA mB,eAAO,cAAP,EAAO,sBAAP,S;UAAA,cAAGB,iB;UA14S/B,IA5HsC,SA4HIC,CAAU,OOAV,EAAiB,OOAJB, CAAJ,C;YAA2C,sBAAI,OOAJ,C;;QA5H/C,OA8HO,W;O;KAtIX,C;oGAWA,6C;MA26SiB,gB;MADb,YAAY,C; MACZ,iD;QAAa,WAAb,e;QA16SI,IAAI,Wak6SkB,cAl6SIB,Eak6SkB,sBA16SIB,Wak6S2B,IA16S3B,CAAJ,C;U AA2C,sBAk6SZ,IA16SY,C;;MAE/C,OOAO,W;K;qGAGX,6C;MAu6SiB,gB;MADb,YAAY,C;MACZ,iD;QAAa,W AAAb,e;QA95SI,IAAI,WA85SkB,cA95SIB,EA85SkB,sBA95SIB,WA85S2B,IA95S3B,CAAJ,C;UAA2C,sBA85SZ,I A95SY,C;;MAE/C,OOAO,W;K;sGAGX,6C;MAm6SiB,gB;MADb,YAAY,C;MACZ,iD;QAAa,WAAb,e;QA15SI,IA AI,WA05SkB,cA15SIB,EA05SkB,sBA15SIB,WA05S2B,IA15S3B,CAAJ,C;UAA2C,sBA05SZ,IA15SY,C;;MAE /C,OOAO,W;K;qGAGX,6C;MA+5SiB,gB;MADb,YAAY,C;MACZ,iD;QAAa,WAAb,e;QA5SI,IAAI,WAs5SkB,c At5SIB,EAs5SkB,sBA5SIB,WAs5S2B,IAt5S3B,CAAJ,C;UAA2C,sBA5SZ,IAt5SY,C;;MAE/C,OOAO,W;K;sGA GX,6C;MA25SiB,gB;MADb,YAAY,C;MACZ,iD;QAAa,WAAb,e;QA15SI,IAAI,Wak5SkB,cAl5SIB,Eak5SkB,sB Al5SIB,Wak5S2B,IA15S3B,CAAJ,C;UAA2C,sBAk5SZ,IA15SY,C;;MAE/C,OOAO,W;K;sGAGX,6C;MAu5SiB,g B;MADb,YAAY,C;MACZ,iD;QAAa,WAAb,e;QA94SI,IAAI,WA84SkB,cA94SIB,EA84SkB,sBA94SIB,WA84S2 B,IA94S3B,CAAJ,C;UAA2C,sBA84SZ,IA94SY,C;;MAE/C,OOAO,W;K;sGAGX,6C;MAm5SiB,gB;MADb,YAA Y,C;MACZ,iD;QAAa,WAAb,e;QA14SI,IAAI,WA04SkB,cA14SIB,EA04SkB,sBA14SIB,WA04S2B,IA14S3B,CA AJ,C;UAA2C,sBA04SZ,IA14SY,C;;MAE/C,OOAO,W;K;sGAGX,6C;MA+4SiB,gB;MADb,YAAY,C;MACZ,iD; QAAa,WAAb,e;QA5SI,IAAI,WAs4SkB,cAt4SIB,EAs4SkB,sBA4SIB,WAs4S2B,IAt4S3B,CAAJ,C;UAA2C,sBA s4SZ,IAt4SY,C;;MAE/C,OOAO,W;K;sGAGX,yB;MAAA,gC;MAo4SA,oC;MAP4SA,oD;QA24SiB,gB;QADb,YA AY,C;QACZ,iD;UAAa,WAAb,0B;UAAmB,eAAO,cAAP,EAAO,sBAAP,S;UAAA,cAAGB,iB;UA14S/B,IAAI,UA AU,OOAV,EAAiB,OOAJB,CAAJ,C;YAA2C,sBAAI,OOAJ,C;;QAE/C,OOAO,W;O;KAXX,C;sGAcA,yB;MAAA, +D;MAAA,sC;QAMW,kBAAmB,gB;QASV,Q;QAahB,iD;UAAgB,cAAhB,e;UAAsB,IAAI,YAAJ,C;YAAkB,WA AY,WAAI,OOAJ,C;;QATpD,OAuO,W;O;KAhBX,C;OGASA,4C;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;QA AgB,cAAA,SAAhB,M;QAAsB,IAAI,YAAJ,C;UAAkB,WAAy,WAAI,OOAJ,C;;MACpD,OOAO,W;K;wFAGX,y B;MAAA,+D;MAAA,uC;QAMW,kBAAY,gB;QAoGH,Q;QAahB,iD;UAAgB,cAAhB,e;UAAsB,IAAI,CAPGS,SA oGR,CAAU,OOAV,CAAL,C;YAAyB,WAAy,WAAI,OOAJ,C;;QApg3D,OAqGO,W;O;KA3GX,C;OFASA,yB;M AAA,+D;MAAA,uC;QAMW,kBAAY,gB;QAqGH,Q;QAahB,iD;UAAgB,cAAhB,e;UAAsB,IAAI,CARGY,SAqG X,CAAU,OOAV,CAAL,C;YAAyB,WAAy,WAAI,OOAJ,C;;QArG3D,OAsGO,W;O;KA5GX,C;OFASA,yB;MAA A,+D;MAAA,uC;QAMW,kBAAY,gB;QAsGH,Q;QAahB,iD;UAAgB,cAAhB,e;UAAsB,IAAI,CAtGa,SAsGZ,CA



uIe,W;ONiuItC,Oe7rIuC,Of6rIhC,yBAAY,OAAQ,MAApB,EAA2B,OAAQ,aAAR,GAAuB,CAAvB,IAA3B,Ce7rI  
gC,C;K;IfgsI3C,qC;MAII,IAAI,OAAQ,UAAZ,C;QAAuB,OMxule,W;ONyuItC,Oe7rIqC,Of6rI9B,yBAAY,OAAQ,  
MAApB,EAA2B,OAAQ,aAAR,GAAuB,CAAvB,IAA3B,Ce7rI8B,C;K;IfgsIzC,qC;MAII,IAAI,OAAQ,UAAZ,C;Q  
AAuB,OMhve,W;ONivItC,Oe7rIsC,Of6rI/B,yBAAY,OAAQ,MAApB,EAA2B,OAAQ,aAAR,GAAuB,CAAvB,IA  
A3B,Ce7rI+B,C;K;IfgsI1C,qC;MAII,IAAI,OAAQ,UAAZ,C;QAAuB,OMxvIe,W;ONyvItC,Oe7rIuC,Of6rIhC,yBA  
AY,OAAQ,MAApB,EAA2B,OAAQ,aAAR,GAAuB,CAAvB,IAA3B,Ce7rIgC,C;K;IfgsI3C,qC;MAII,IAAI,OAAQ,  
UAAZ,C;QAAuB,OMhwe,W;ONiwItC,Oe7rIwC,Of6rIjC,yBAAY,OAAQ,MAApB,EAA2B,OAAQ,aAAR,GAAu  
B,CAAvB,IAA3B,Ce7rIiC,C;K;IfgsI5C,qC;MAII,IAAI,OAAQ,UAAZ,C;QAAuB,OMxwIe,W;ONywItC,Oe7rIyC,  
Of6rIiC,0BAAY,OAAQ,MAApB,EAA2B,OAAQ,aAAR,GAAuB,CAAvB,IAA3B,Ce7rIkC,C;K;IfgsI7C,qC;MAII,  
IAAI,OAAQ,UAAZ,C;QAAuB,OMhxIe,W;ONixItC,OAA4D,SAArD,0BAAY,OAAQ,MAApB,EAA2B,OAAQ,a  
AAR,GAAuB,CAAvB,IAA3B,CAAqD,C;K;IAGhE,qC;MAOkB,Q;MAHd,WAAmB,wBAAR,OAAQ,EAAwB,EA  
AxB,C;MACnB,IAAI,SAAQ,CAAZ,C;QAAe,OAAO,W;MACTb,WAAW,iBAaA,IAAb,C;MACG,yB;MAAd,OA  
Ac,cAAd,C;QAAC,uB;QACV,IAAK,WAAI,UAAI,KAAJ,CAAJ,C;;MAET,OAAO,I;K;IAGX,qC;MAOkB,Q;MAH  
d,WAAmB,wBAAR,OAAQ,EAAwB,EAAXB,C;MACnB,IAAI,SAAQ,CAAZ,C;QAAe,OAAO,W;MACTb,WAA  
W,iBAAGB,IAAhB,C;MACG,yB;MAAd,OAAC,cAAd,C;QAAC,uB;QACV,IAAK,WAAI,UAAI,KAAJ,CAAJ,C;;  
MAET,OAAO,I;K;IAGX,sC;MAOkB,Q;MAHd,WAAmB,wBAAR,OAAQ,EAAwB,EAAXB,C;MACnB,IAAI,SA  
AQ,CAAZ,C;QAAe,OAAO,W;MACTb,WAAW,iBAaIB,IAAjB,C;MACG,yB;MAAd,OAAC,cAAd,C;QAAC,uB;Q  
ACV,IAAK,WAAI,UAAI,KAAJ,CAAJ,C;;MAET,OAAO,I;K;IAGX,sC;MAOkB,Q;MAHd,WAAmB,wBAAR,OA  
AQ,EAAwB,EAAXB,C;MACnB,IAAI,SAAQ,CAAZ,C;QAAe,OAAO,W;MACTb,WAAW,iBAaE,IAAf,C;MACG,  
yB;MAAd,OAAC,cAAd,C;QAAC,uB;QACV,IAAK,WAAI,UAAI,KAAJ,CAAJ,C;;MAET,OAAO,I;K;IAGX,sC;M  
AOkB,Q;MAHd,WAAmB,wBAAR,OAAQ,EAAwB,EAAXB,C;MACnB,IAAI,SAAQ,CAAZ,C;QAAe,OAAO,W;  
MACTb,WAAW,iBAAGB,IAAhB,C;MACG,yB;MAAd,OAAC,cAAd,C;QAAC,uB;QACV,IAAK,WAAI,UAAI,KA  
AJ,CAAJ,C;;MAET,OAAO,I;K;IAGX,sC;MAOkB,Q;MAHd,WAAmB,wBAAR,OAAQ,EAAwB,EAAXB,C;MAC  
nB,IAAI,SAAQ,CAAZ,C;QAAe,OAAO,W;MACTb,WAAW,iBAaIB,IAAjB,C;MACG,yB;MAAd,OAAC,cAAd,C;  
QAAC,uB;QACV,IAAK,WAAI,UAAI,KAAJ,CAAJ,C;;MAET,OAAO,I;K;IAGX,sC;MAOkB,Q;MAHd,WAAmB,  
wBAAR,OAAQ,EAAwB,EAAXB,C;MACnB,IAAI,SAAQ,CAAZ,C;QAAe,OAAO,W;MACTb,WAAW,iBAaKB,IA  
AIB,C;MACG,yB;MAAd,OAAC,cAAd,C;QAAC,uB;QACV,IAAK,WAAI,UAAI,KAAJ,CAAJ,C;;MAET,OAAO,  
I;K;IAGX,sC;MAOkB,Q;MAHd,WAAmB,wBAAR,OAAQ,EAAwB,EAAXB,C;MACnB,IAAI,SAAQ,CAAZ,C;Q  
AAe,OAAO,W;MACTb,WAAW,iBAaMB,IAAnB,C;MACG,yB;MAAd,OAAC,cAAd,C;QAAC,uB;QACV,IAAK,  
WAAI,UAAI,KAAJ,CAAJ,C;;MAET,OAAO,I;K;IAGX,sC;MAOkB,Q;MAHd,WAAmB,wBAAR,OAAQ,EAAwB  
,EAAXB,C;MACnB,IAAI,SAAQ,CAAZ,C;QAAe,OAAO,W;MACTb,WAAW,iBAAGB,IAAhB,C;MACG,yB;MA  
Ad,OAAC,cAAd,C;QAAC,uB;QACV,IAAK,WAAI,sBAI,KAAJ,EAJ,C;;MAET,OAAO,I;K;IAGX,wC;MAMw  
B,UACTION;MAHX,aAAa,aAAa,SAAb,EAAMB,OAAQ,KAA3B,C;MACb,kBAaKB,C;MACE,yB;MAApB,OAAO  
B,cAApB,C;QAAoB,6B;QACHB,OAAO,oBAAP,EAAO,4BAAP,YAAwB,UAAK,WAAL,C;;MAE5B,OAAO,M;  
K;IAGX,0C;MAMwB,UACTION;MAHX,aAAa,cAAU,OAAQ,KAAIB,C;MACb,kBAaKB,C;MACE,yB;MAApB,OA  
AoB,cAApB,C;QAAoB,6B;QACHB,OAAO,oBAAP,EAAO,4BAAP,YAAwB,UAAK,WAAL,C;;MAE5B,OAAO  
,M;K;IAGX,0C;MAMwB,UACTION;MAHX,aAAa,eAAW,OAAQ,KAAIB,C;MACb,kBAaKB,C;MACE,yB;MAA  
pB,OAAoB,cAApB,C;QAAoB,6B;QACHB,OAAO,oBAAP,EAAO,4BAAP,YAAwB,UAAK,WAAL,C;;MAE5B,OA  
AO,M;K;IAGX,0C;MAMwB,UACTION;MAHX,aAAa,iBAAU,OAAQ,KAAIB,C;MACb,kBAaKB,C;MACE,  
yB;MAApB,OAAoB,cAApB,C;QAAoB,6B;QACHB,OAAO,oBAAP,EAAO,4BAAP,YAAwB,UAAK,WAAL,C;;  
MAE5B,OAAO,M;K;IAGX,0C;MAMwB,UACTION;MAHX,aAAa,iBAAW,OAAQ,KAAIB,C;MACb,kBAaKB,C;  
MACE,yB;MAApB,OAAoB,cAApB,C;QAAoB,6B;QACHB,OAAO,oBAAP,EAAO,4BAAP,YAAwB,UAAK,WA  
AL,C;;MAE5B,OAAO,M;K;IAGX,0C;MAMwB,UACTION;MAHX,aAAa,iBAAY,OAAQ,KAApB,C;MACb,kBAA  
KB,C;MACE,yB;MAApB,OAAoB,cAApB,C;QAAoB,6B;QACHB,OAAO,oBAAP,EAAO,4BAAP,YAAwB,UAAK  
,WAAL,C;;MAE5B,OAAO,M;K;IAGX,0C;MAMwB,UACTION;MAHX,aAAa,oBAaA,OAAQ,KAArB,C;MACb,k  
BAaKB,C;MACE,yB;MAApB,OAAoB,cAApB,C;QAAoB,6B;QACHB,OAAO,oBAAP,EAAO,4BAAP,YAAwB,U

AAK,WAAL,C;;MAE5B,OAAO,M;K;IAGX,0C;MAMwB,UACT,M;MAHX,aAAa,iBAAU,OAAQ,KAAIB,C;MA  
Cb,kBAaKb,C;MACE,yB;MAApB,OAAoB,cAApB,C;QAAoB,6B;QACbB,OAAO,oBAAP,EAAO,4BAAP,YAA  
wB,UAAK,WAAL,C;;MAE5B,OAAO,M;K;IAGX,0C;MAII,IAAI,OAAQ,UAAZ,C;QAAuB,OAAO,yBAAY,CAA  
Z,EAAe,CAAf,C;MAC9B,OAAO,yBAAY,OAAQ,MAApB,EAA2B,OAAQ,aAAR,GAAuB,CAAvB,IAA3B,C;K;I  
AGX,0C;MAII,IAAI,OAAQ,UAAZ,C;QAAuB,OAAO,cAAU,CAAV,C;MAC9B,OAAO,yBAAY,OAAQ,MAApB,  
EAA2B,OAAQ,aAAR,GAAuB,CAAvB,IAA3B,C;K;IAGX,2C;MAII,IAAI,OAAQ,UAAZ,C;QAAuB,OAAO,eAA  
W,CAAX,C;MAC9B,OAAO,yBAAY,OAAQ,MAApB,EAA2B,OAAQ,aAAR,GAAuB,CAAvB,IAA3B,C;K;IAGX  
,2C;MAII,IAAI,OAAQ,UAAZ,C;QAAuB,OAAO,eAAS,CAAT,C;MAC9B,OAAO,yBAAY,OAAQ,MAApB,EAA  
2B,OAAQ,aAAR,GAAuB,CAAvB,IAA3B,C;K;IAGX,2C;MAII,IAAI,OAAQ,UAAZ,C;QAAuB,OAAO,iBAAU,C  
AAV,C;MAC9B,OAAO,yBAAY,OAAQ,MAApB,EAA2B,OAAQ,aAAR,GAAuB,CAAvB,IAA3B,C;K;IAGX,2C;  
MAII,IAAI,OAAQ,UAAZ,C;QAAuB,OAAO,iBAAW,CAAX,C;MAC9B,OAAO,yBAAY,OAAQ,MAApB,EAA2  
B,OAAQ,aAAR,GAAuB,CAAvB,IAA3B,C;K;IAGX,2C;MAII,IAAI,OAAQ,UAAZ,C;QAAuB,OAAO,iBAAAY,CA  
AZ,C;MAC9B,OAAO,yBAAY,OAAQ,MAApB,EAA2B,OAAQ,aAAR,GAAuB,CAAvB,IAA3B,C;K;IAGX,2C;M  
AII,IAAI,OAAQ,UAAZ,C;QAAuB,OAAO,oBAAa,CAAb,C;MAC9B,OAAO,0BAAY,OAAQ,MAApB,EAA2B,O  
AAQ,aAAR,GAAuB,CAAvB,IAA3B,C;K;IAGX,2C;MAII,IAAI,OAAQ,UAAZ,C;QAAuB,OAAO,iBAAU,CAAV  
,C;MAC9B,OAAO,0BAAY,OAAQ,MAApB,EAA2B,OAAQ,aAAR,GAAuB,CAAvB,IAA3B,C;K;IAGX,4B;MAci  
B,Q;Mc3nJb,IAAI,EdqnJI,KAAK,CcrnJT,CAAJ,C;QACI,cdonJc,sD;QcnnJd,MAAM,gCAAYB,OAAQ,WAAjC,C;  
OdonJV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,IAAI,KAAK,gBAAT,C;QAAe,OAAO,iB;MAcIB,IAA  
I,MAAK,CAAT,C;QAAY,OAAO,OAAO,UAAK,CAAL,CAAP,C;MACnB,YAAY,C;MACZ,WAAW,iBAAa,CA  
Ab,C;MACX,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,IAAK,WAAI,IAAJ,C;QACL,IAAI,mCAAW,CA  
Af,C;UACI,K;;MAER,OAAO,I;K;IAGX,8B;MAciB,Q;McyjJb,IAAI,Ed2oJI,KAAK,Cc3oJT,CAAJ,C;QACI,cd0oJ  
c,sD;QczoJd,MAAM,gCAAYB,OAAQ,WAAjC,C;Od0oJV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,IAA  
I,KAAK,gBAAT,C;QAAe,OAAO,mB;MAcIB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,UAAK,CAAL,CAA  
P,C;MACnB,YAAY,C;MACZ,WAAW,iBAAgB,CAAhB,C;MACX,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;Q  
ACI,IAAK,WAAI,IAAJ,C;QACL,IAAI,mCAAW,CAAf,C;UACI,K;;MAER,OAAO,I;K;IAGX,8B;MAciB,Q;Mcvq  
Jb,IAAI,EdiqJI,KAAK,CcjqtJ,CAAJ,C;QACI,cdgqJc,sD;Qc/pJd,MAAM,gCAAYB,OAAQ,WAAjC,C;OdgqJV,IA  
AI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,IAAI,KAAK,gBAAT,C;QAAe,OAAO,mB;MAcIB,IAAI,MAAK,  
CAAT,C;QAAY,OAAO,OAAO,UAAK,CAAL,CAAP,C;MACnB,YAAY,C;MACZ,WAAW,iBAAiB,CAAjB,C;M  
ACX,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,IAAK,WAAI,IAAJ,C;QACL,IAAI,mCAAW,CAAf,C;UA  
CI,K;;MAER,OAAO,I;K;IAGX,8B;MAciB,Q;MczuJb,IAAI,EdmuJI,KAAK,CcnuJT,CAAJ,C;QACI,cdkuJc,sD;QcjuJd,MAAM,gC  
AAyB,OAAQ,WAAjC,C;OdkuJV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,IAAI,KAAK,gBAAT,C;QA  
Ae,OAAO,mB;MAcIB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,UAAK,CAAL,CAAP,C;MACnB,YAAY,C;  
MACZ,WAAW,iBAAiB,CAAjB,C;MACX,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,IAAK,WAAI,IAAJ,  
C;QACL,IAAI,mCAAW,CAAf,C;UACI,K;;MAER,OAAO,I;K;IAGX,8B;MAciB,Q;MentJb,IAAI,Ed6sJI,  
KAAK,Cc7sJT,CAAJ,C;QACI,cd4sJc,sD;Qc3sJd,MAAM,gCAAYB,OAAQ,WAAjC,C;Od4sJV,IAAI,MAAK,CAA  
T,C;QAAY,OAAO,W;MACnB,IAAI,KAAK,gBAAT,C;QAAe,OAAO,mB;MAcIB,IAAI,MAAK,CAAT,C;QAAY  
,OAAO,OAAO,UAAK,CAAL,CAAP,C;MACnB,YAAY,C;MACZ,WAAW,iBAAgB,CAAhB,C;MACX,wBAAa,S  
AAb,gB;QAAa,WAAA,SAAb,M;QACI,IAAK,WAAI,IAAJ,C;QACL,IAAI,mCAAW,CAAf,C;UACI,K;;MAER,O  
AAO,I;K;IAGX,8B;MAciB,Q;MczuJb,IAAI,EdmuJI,KAAK,CcnuJT,CAAJ,C;QACI,cdkuJc,sD;QcjuJd,MAAM,gC  
AAyB,OAAQ,WAAjC,C;OdkuJV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,IAAI,KAAK,gBAAT,C;QA  
Ae,OAAO,mB;MAcIB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,UAAK,CAAL,CAAP,C;MACnB,YAAY,C;  
MACZ,WAAW,iBAAiB,CAAjB,C;MACX,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,IAAK,WAAI,IAAJ,  
C;QACL,IAAI,mCAAW,CAAf,C;UACI,K;;MAER,OAAO,I;K;IAGX,8B;MAciB,Q;McvJb,IAAI,EdyvJI,KAAK,  
CczvJT,CAAJ,C;QACI,cdwvJc,sD;QcqvJd,MAAM,gCAAYB,OAAQ,WAAjC,C;OdwvJV,IAAI,MAAK,CAAT,C;  
QAAY,OAAO,W;MACnB,IAAI,KAAK,gBAAT,C;QAAe,OAAO,mB;MAcIB,IAAI,MAAK,CAAT,C;QAAY,OA  
AO,OAAO,UAAK,CAAL,CAAP,C;MACnB,YAAY,C;MACZ,WAAW,iBAAkB,CAAlB,C;MACX,wBAAa,SAAb  
,gB;QAAa,WAAA,SAAb,M;QACI,IAAK,WAAI,IAAJ,C;QACL,IAAI,mCAAW,CAAf,C;UACI,K;;MAER,OAAO  
,I;K;IAGX,8B;MAciB,Q;McrxJb,IAAI,Ed+wJI,KAAK,Cc/wJT,CAAJ,C;QACI,cd8wJc,sD;Qc7wJd,MAAM,gCAA

yB,OAAQ,WAAjC,C;Od8wJV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,IAAI,KAAK,gBAAT,C;QA Ae,  
OAAO,mB;MACTB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,UAAK,CAAL,CAAP,C;MACnB,YAAY,C;MA  
CZ,WAAW,iBAAmB,CAAnB,C;MACX,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,IAAK,WAAI,IAAJ,C;  
QACL,IAAI,mCAAW,CAAf,C;UACI,K;;MAER,OAAO,I;K;IAGX,8B;MAciB,Q;Mc3yJb,IAAI,EdqyJI,KAAK,Cc  
ryJT,CAAJ,C;QACI,cdoyJc,sD;QcnyJd,MAAM,gCAAYB,OAAQ,WAAjC,C;OdoyJV,IAAI,MAAK,CAAT,C;QAA  
Y,OAAO,W;MACnB,IAAI,KAAK,gBAAT,C;QA Ae,OAAO,mB;MACTB,IAAI,MAAK,CAAT,C;QAAY,OAAO,O  
AAO,sBAAK,CAAL,EAAP,C;MACnB,YAAY,C;MACZ,WAAW,iBAAgB,CAAhB,C;MACX,wBAAa,SAAb,gB;  
QAAa,WAAb,UAAa,SAAb,O;QACI,IAAK,WAAI,iBAAJ,C;QACL,IAAI,mCAAW,CAAf,C;UACI,K;;MAER,OA  
AO,I;K;IAGX,gC;MenzJI,IAAI,Ed2zJI,KAAK,Cc3zJT,CAAJ,C;QACI,cd0zJc,sD;QczzJd,MAAM,gCAAYB,OAA  
Q,WAAjC,C;Od0zJV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,WAAW,gB;MACX,IAAI,KAAK,IAAT,  
C;QA Ae,OAAO,iB;MACTB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,UAAK,OAAO,CAAP,IAAL,CAAP,C;  
MACnB,WAAW,iBAaA,CAAb,C;MACX,iBAAc,OAAO,CAAP,IAAd,UAA6B,IAA7B,U;QACI,IAAK,WAAI,UA  
AK,KAAL,CAAJ,C;MACT,OAAO,I;K;IAGX,kC;Mct0JI,IAAI,Ed80JI,KAAK,Cc90JT,CAAJ,C;QACI,cd60Jc,sD;  
Qc50Jd,MAAM,gCAAYB,OAAQ,WAAjC,C;Od60JV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,WAAW,  
gB;MACX,IAAI,KAAK,IAAT,C;QA Ae,OAAO,mB;MACTB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,UAA  
K,OAAO,CAAP,IAAL,CAAP,C;MACnB,WAAW,iBAAgB,CAAhB,C;MACX,iBAAc,OAAO,CAAP,IAAd,UAA6  
B,IAA7B,U;QACI,IAAK,WAAI,UAAK,KAAL,CAAJ,C;MACT,OAAO,I;K;IAGX,kC;Mcz1JI,IAAI,Edi2JI,KAA  
K,Ccj2JT,CAAJ,C;QACI,cdg2Jc,sD;Qc/1Jd,MAAM,gCAAYB,OAAQ,WAAjC,C;Odg2JV,IAAI,MAAK,CAAT,C;  
QAAY,OAAO,W;MACnB,WAAW,gB;MACX,IAAI,KAAK,IAAT,C;QA Ae,OAAO,mB;MACTB,IAAI,MAAK,CA  
AT,C;QAAY,OAAO,OAAO,UAAK,OAAO,CAAP,IAAL,CAAP,C;MACnB,WAAW,iBAAiB,CAAjB,C;MACX,i  
BAAc,OAAO,CAAP,IAAd,UAA6B,IAA7B,U;QACI,IAAK,WAAI,UAAK,KAAL,CAAJ,C;MACT,OAAO,I;K;IA  
GX,kC;Mc52JI,IAAI,Edo3JI,KAAK,Ccp3JT,CAAJ,C;QACI,cdm3Jc,sD;Qcl3Jd,MAAM,gCAAYB,OAAQ,WAAjC,  
C;Odm3JV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,WAAW,gB;MACX,IAAI,KAAK,IAAT,C;QA Ae,O  
AAO,mB;MACTB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,UAAK,OAAO,CAAP,IAAL,CAAP,C;MACnB,  
WAAW,iBA Ae,CAAf,C;MACX,iBAAc,OAAO,CAAP,IAAd,UAA6B,IAA7B,U;QACI,IAAK,WAAI,UAAK,KAA  
L,CAAJ,C;MACT,OAAO,I;K;IAGX,kC;Mc/3JI,IAAI,Edu4JI,KAAK,Ccv4JT,CAAJ,C;QACI,cds4Jc,sD;Qcr4Jd,M  
AAM,gCAAYB,OAAQ,WAAjC,C;Ods4JV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,WAAW,gB;MACX  
,IAAI,KAAK,IAAT,C;QA Ae,OAAO,mB;MACTB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,UAAK,OAAO,C  
AAP,IAAL,CAAP,C;MACnB,WAAW,iBAAgB,CAAhB,C;MACX,iBAAc,OAAO,CAAP,IAAd,UAA6B,IAA7B,U  
;QACI,IAAK,WAAI,UAAK,KAAL,CAAJ,C;MACT,OAAO,I;K;IAGX,kC;Mcl5JI,IAAI,Ed05JI,KAAK,Cc15JT,C  
AAJ,C;QACI,cdy5Jc,sD;Qcx5Jd,MAAM,gCAAYB,OAAQ,WAAjC,C;Ody5JV,IAAI,MAAK,CAAT,C;QAAY,OA  
AO,W;MACnB,WAAW,gB;MACX,IAAI,KAAK,IAAT,C;QA Ae,OAAO,mB;MACTB,IAAI,MAAK,CAAT,C;QA  
AY,OAAO,OAAO,UAAK,OAAO,CAAP,IAAL,CAAP,C;MACnB,WAAW,iBAAiB,CAAjB,C;MACX,iBAAc,OA  
AO,CAAP,IAAd,UAA6B,IAA7B,U;QACI,IAAK,WAAI,UAAK,KAAL,CAAJ,C;MACT,OAAO,I;K;IAGX,kC;Mc  
r6JI,IAAI,Ed66JI,KAAK,Cc76JT,CAAJ,C;QACI,cd46Jc,sD;Qc36Jd,MAAM,gCAAYB,OAAQ,WAAjC,C;Od46JV,  
IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,WAAW,gB;MACX,IAAI,KAAK,IAAT,C;QA Ae,OAAO,mB;M  
ACTB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,UAAK,OAAO,CAAP,IAAL,CAAP,C;MACnB,WAAW,iBA  
AkB,CAAlB,C;MACX,iBAAc,OAAO,CAAP,IAAd,UAA6B,IAA7B,U;QACI,IAAK,WAAI,UAAK,KAAL,CAAJ,  
C;MACT,OAAO,I;K;IAGX,kC;Mcx7JI,IAAI,Edg8JI,KAAK,Cch8JT,CAAJ,C;QACI,cd+7Jc,sD;Qc97Jd,MAAM,g  
CAAYB,OAAQ,WAAjC,C;Od+7JV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,WAAW,gB;MACX,IAAI,  
KAAK,IAAT,C;QA Ae,OAAO,mB;MACTB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,UAAK,OAAO,CAAP,I  
AAL,CAAP,C;MACnB,WAAW,iBAAmB,CAAnB,C;MACX,iBAAc,OAAO,CAAP,IAAd,UAA6B,IAA7B,U;QA  
CI,IAAK,WAAI,UAAK,KAAL,CAAJ,C;MACT,OAAO,I;K;IAGX,kC;Mc38JI,IAAI,Edm9JI,KAAK,Ccn9JT,CAA  
J,C;QACI,cdk9Jc,sD;Qcj9Jd,MAAM,gCAAYB,OAAQ,WAAjC,C;Odk9JV,IAAI,MAAK,CAAT,C;QAAY,OAAO,  
W;MACnB,WAAW,gB;MACX,IAAI,KAAK,IAAT,C;QA Ae,OAAO,mB;MACTB,IAAI,MAAK,CAAT,C;QAAY,  
OAAO,OAAO,sBAAK,OAAO,CAAP,IAAL,EAAP,C;MACnB,WAAW,iBAAgB,CAAhB,C;MACX,iBAAc,OAA  
O,CAAP,IAAd,UAA6B,IAA7B,U;QACI,IAAK,WAAI,sBAAK,KAAL,EAAP,C;MACT,OAAO,I;K;gGAGX,yB;M  
AAA,8D;MAAA,4C;MAAA,gD;MAAA,uC;QAMI,iBAAc,wBAAd,WAA+B,CAA/B,U;UACI,IAAI,CAAC,UAA





ACN,WAAL,IAAK,C;MACL,OAAO,I;K;IAGX,+B;MAIL,IAqnEO,qBAAQ,CArnEf,C;QAAe,OAAO,W;MACTb,WAAW,0B;MACN,WAAL,IAAK,C;MACL,OAAO,I;K;IAGX,+B;MAIL,IAmnEO,qBAAQ,CAnnEf,C;QAAe,OA AO,W;MACTb,WAAW,0B;MACN,WAAL,IAAK,C;MACL,OAAO,I;K;IAGX,+B;MAIL,IAinEO,qBAAQ,CAjnEf, C;QAAe,OAAO,W;MACTb,WAAW,0B;MACN,WAAL,IAAK,C;MACL,OAAO,I;K;IAGX,+B;MAIL,IA+mEO,qB AAQ,CA/mEf,C;QAAe,OAAO,W;MACTb,WAAW,0B;MACN,WAAL,IAAK,C;MACL,OAAO,I;K;IAGX,kC;MA II,IAqiEO,qBAAQ,CARiEf,C;QAAe,OAAO,S;MACTb,aAAa,aAAa,SAAb,EAAmB,gBAAnB,C;MACb,gBAAgB,w B;MACHb,aAAU,CAAV,OAAa,SAAb,M;QACI,OAAO,YAAY,CAAZ,IAAP,IAAwB,UAAK,CAAL,C;MAC5B,O AAO,M;K;IAGX,oC;MAIL,IAiiEO,qBAAQ,CAjiEf,C;QAAe,OAAO,S;MACTb,aAAa,cAAU,gBAAV,C;MACb,gB AAgB,0B;MACHb,aAAU,CAAV,OAAa,SAAb,M;QACI,OAAO,YAAY,CAAZ,IAAP,IAAwB,UAAK,CAAL,C;M AC5B,OAAO,M;K;IAGX,oC;MAIL,IA6hEO,qBAAQ,CA7hEf,C;QAAe,OAAO,S;MACTb,aAAa,eAAW,gBAAX, C;MACb,gBAAgB,0B;MACHb,aAAU,CAAV,OAAa,SAAb,M;QACI,OAAO,YAAY,CAAZ,IAAP,IAAwB,UAAK ,CAAL,C;MAC5B,OAAO,M;K;IAGX,oC;MAIL,IAyhEO,qBAAQ,CAzhEf,C;QAAe,OAAO,S;MACTb,aAAa,eAA S,gBAAT,C;MACb,gBAAgB,0B;MACHb,aAAU,CAAV,OAAa,SAAb,M;QACI,OAAO,YAAY,CAAZ,IAAP,IAA wB,UAAK,CAAL,C;MAC5B,OAAO,M;K;IAGX,oC;MAIL,IAqhEO,qBAAQ,CArhEf,C;QAAe,OAAO,S;MACTb,a AAa,iBAAU,gBAAV,C;MACb,gBAAgB,0B;MACHb,aAAU,CAAV,OAAa,SAAb,M;QACI,OAAO,YAAY,CAAZ ,IAAP,IAAwB,UAAK,CAAL,C;MAC5B,OAAO,M;K;IAGX,oC;MAIL,IAihEO,qBAAQ,CAjhEf,C;QAAe,OAAO, S;MACTb,aAAa,iBAAW,gBAAX,C;MACb,gBAAgB,0B;MACHb,aAAU,CAAV,OAAa,SAAb,M;QACI,OAAO,Y AAY,CAAZ,IAAP,IAAwB,UAAK,CAAL,C;MAC5B,OAAO,M;K;IAGX,oC;MAIL,IA6gEO,qBAAQ,CA7gEf,C;Q AAe,OAAO,S;MACTb,aAAa,iBAAy,gBAAZ,C;MACb,gBAAgB,0B;MACHb,aAAU,CAAV,OAAa,SAAb,M;QA CI,OAAO,YAAY,CAAZ,IAAP,IAAwB,UAAK,CAAL,C;MAC5B,OAAO,M;K;IAGX,oC;MAIL,IAygEO,qBAAQ, CAzgEf,C;QAAe,OAAO,S;MACTb,aAAa,oBAAa,gBAAb,C;MACb,gBAAgB,0B;MACHb,aAAU,CAAV,OAAa,S AAb,M;QACI,OAAO,YAAY,CAAZ,IAAP,IAAwB,UAAK,CAAL,C;MAC5B,OAAO,M;K;IAGX,oC;MAIL,IAqgE O,qBAAQ,CArgEf,C;QAAe,OAAO,S;MACTb,aAAa,iBAAU,gBAAV,C;MACb,gBAAgB,0B;MACHb,aAAU,CA AV,OAAa,SAAb,M;QACI,OAAO,YAAY,CAAZ,IAAP,IAAwB,UAAK,CAAL,C;MAC5B,OAAO,M;K;IAGX,4B; MAKI,qBAAQ,4BAAR,C;K;IAGJ,8B;MAKI,qBAAQ,4BAAR,C;K;IAGJ,8B;MAKI,sBAAQ,4BAAR,C;K;IAGJ,8 B;MAKI,sBAAQ,4BAAR,C;K;IAGJ,8B;MAKI,sBAAQ,4BAAR,C;K;IAGJ,8B;MAKI,sBAAQ,4BAAR,C;K;IAG J,8B;MAKI,sBAAQ,4BAAR,C;K;IAGJ,8B;MAKI,sBAAQ,4BAAR,C;K;IAGJ,8B;MAKI,sBAAQ,4BAAR,C;K;IAG J,sC;MAOI,aAAU,wBAAV,OAA2B,CAA3B,M;QACI,QAAQ,MAAO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,WAA W,UAAK,CAAL,C;QACX,UAAK,CAAL,IAAU,UAAK,CAAL,C;QACV,UAAK,CAAL,IAAU,I;K;IAIIB,sC;MA OI,aAAU,0BAAV,OAA2B,CAA3B,M;QACI,QAAQ,MAAO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,WAAW,UAAK, CAAL,C;QACX,UAAK,CAAL,IAAU,UAAK,CAAL,C;QACV,UAAK,CAAL,IAAU,I;K;IAIIB,uC;MAOI,aAAU, 0BAAV,OAA2B,CAA3B,M;QACI,QAAQ,MAAO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,WAAW,UAAK,CAAL,C; QACX,UAAK,CAAL,IAAU,UAAK,CAAL,C;QACV,UAAK,CAAL,IAAU,I;K;IAIIB,uC;MAOI,aAAU,0BAAV, OAA2B,CAA3B,M;QACI,QAAQ,MAAO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,WAAW,UAAK,CAAL,C;QACX,U AAK,CAAL,IAAU,UAAK,CAAL,C;QACV,UAAK,CAAL,IAAU,I;K;IAIIB,uC;MAOI,aAAU,0BAAV,OAA2B,C AA3B,M;QACI,QAAQ,MAAO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,WAAW,UAAK,CAAL,C;QACX,UAAK,CA AL,IAAU,UAAK,CAAL,C;QACV,UAAK,CAAL,IAAU,I;K;IAIIB,uC;MAOI,aAAU,0BAAV,OAA2B,CAA3B,M ;QACI,QAAQ,MAAO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,WAAW,UAAK,CAAL,C;QACX,UAAK,CAAL,IAAU, UAAK,CAAL,C;QACV,UAAK,CAAL,IAAU,I;K;IAIIB,uC;MAOI,aAAU,0BAAV,OAA2B,CAA3B,M;QACI,QA AQ,MAAO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,WAAW,UAAK,CAAL,C;QACX,UAAK,CAAL,IAAU,UAAK,C AAL,C;QACV,UAAK,CAAL,IAAU,I;K;IAIIB,uC;MAOI,aAAU,0BAAV,OAA2B,CAA3B,M;QACI,QAAQ,MA AO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,WAAW,UAAK,CAAL,C;QACX,UAAK,CAAL,IAAU,UAAK,CAAL,C; QACV,UAAK,CAAL,IAAU,I;K;IAIIB,uC;MAOI,aAAU,0BAAV,OAA2B,CAA3B,M;QACI,QAAQ,MAAO,iB A AQ,IAAI,CAAJ,IAAR,C;QACf,WAAW,UAAK,CAAL,C;QACX,UAAK,CAAL,IAAU,UAAK,CAAL,C;QACV,U AAK,CAAL,IAAU,I;K;IAIIB,yB;MAAA,oD;MgBn5LA,sC;MAAA,oC;MAAA,uBAOe,yB;QArEf,8D;eAqEe,4 B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA5Dd,cAAc,SA4DgB,CA5DhB,CAAd,EAA2B,SA4DM,CA5DN,CAA3 B,C;W;S;OA4DI,C;MhB44Lf,sC;QAMI,IAAI,mBAAO,CAAX,C;UAAc,oBgB15Ld,eAAW,iBhBk5LsB,QgB15LtB, CAAX,ChBk5Lc,C;U;KANIB,C;sGASA,yB;MAAA,oD;MgBz4LA,sC;MAAA,oC;MAAA,iCAOe,yB;QAxFf,8D;e



AAA,uB;YAAU,eAAsB,gB;YAAtB,OA5Dd,cAAc,SA4DgB,CA5DhB,CAAd,EAA2B,SA4DM,CA5DN,CAA3B,C;W;S;OA4DI,C;MhBouMf,sC;QAMI,OAAO,sBgB1uMP,eAAW,iBhB0uMiB,QgB1uMjB,CAAX,ChB0uMO,C;O;KANX,C;wFASA,yB;MAAA,wD;MgBpvMA,sC;MAAA,oC;MAAA,uBAOe,yB;QArEf,8D;eAqEe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA5Dd,cAAc,SA4DgB,CA5DhB,CAAd,EAA2B,SA4DM,CA5DN,CAA3B,C;W;S;OA4DI,C;MhB6uMf,sC;QAMI,OAAO,sBgBnvMP,eAAW,iBhBmvMiB,QgBnvMjB,CAAX,ChBmvMO,C;O;KANX,C;wFASA,yB;MAAA,wD;MgB7vMA,sC;MAAA,oC;MAAA,uBAOe,yB;QArEf,8D;eAqEe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA5Dd,cAAc,SA4DgB,CA5DhB,CAAd,EAA2B,SA4DM,CA5DN,CAA3B,C;W;S;OA4DI,C;MhBsvMf,sC;QAMI,OAAO,sBgB5vMP,eAAW,iBhB4vMiB,QgB5vMjB,CAAX,ChB4vMO,C;O;KANX,C;wFASA,yB;MAAA,wD;MgBtwMA,sC;MAAA,oC;MAAA,uBAOe,yB;QArEf,8D;eAqEe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA5Dd,cAAc,SA4DgB,CA5DhB,CAAd,EAA2B,SA4DM,CA5DN,CAA3B,C;W;S;OA4DI,C;MhB+vMf,sC;QAMI,OAAO,sBgBrwMP,eAAW,iBhBqwMiB,QgBrwMjB,CAAX,ChBqwMO,C;O;KANX,C;0GASA,yB;MAAA,wD;MgB5vMA,sC;MAAA,oC;MAAA,iCAOe,yB;QAxFf,8D;eAwFe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA/Ed,cAAc,SA+EgB,CA/EhB,CAAd,EAA2B,SA+EM,CA/EN,CAA3B,C;W;S;OA+EI,C;MhBqvMf,sC;QAMI,OAAO,sBgB3vMP,eAAW,2BhB2vM2B,QgB3vM3B,CAAX,ChB2vMO,C;O;KANX,C;4GASA,yB;MAAA,wD;MgBrwMA,sC;MAAA,oC;MAAA,iCAOe,yB;QAxFf,8D;eAwFe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA/Ed,cAAc,SA+EgB,CA/EhB,CAAd,EAA2B,SA+EM,CA/EN,CAA3B,C;W;S;OA+EI,C;MhB8vMf,sC;QAI,OAAO,sBgBlwMP,eAAW,2BhBkwM2B,QgBlwM3B,CAAX,ChBkwMO,C;O;KAJX,C;4GAOA,yB;MAAA,wD;MgB5wMA,sC;MAAA,oC;MAAA,iCAOe,yB;QAxFf,8D;eAwFe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA/Ed,cAAc,SA+EgB,CA/EhB,CAAd,EAA2B,SA+EM,CA/EN,CAA3B,C;W;S;OA+EI,C;MhBqwMf,sC;QAI,OAAO,sBgBzwMP,eAAW,2BhBywM2B,QgBzwM3B,CAAX,ChBywMO,C;O;KAJX,C;4GAOA,yB;MAAA,wD;MgBnxMA,sC;MAAA,oC;MAAA,iCAOe,yB;QAxFf,8D;eAwFe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA/Ed,cAAc,SA+EgB,CA/EhB,CAAd,EAA2B,SA+EM,CA/EN,CAA3B,C;W;S;OA+EI,C;MhB4wMf,sC;QAI,OAAO,sBgBhxMP,eAAW,2BhBgxM2B,QgBhxM3B,CAAX,ChBgxMO,C;O;KAJX,C;4GAOA,yB;MAAA,wD;MgB1xMA,sC;MAAA,oC;MAAA,iCAOe,yB;QAxFf,8D;eAwFe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA/Ed,cAAc,SA+EgB,CA/EhB,CAAd,EAA2B,SA+EM,CA/EN,CAA3B,C;W;S;OA+EI,C;MhBmxMf,sC;QAI,OAAO,sBgBvxMP,eAAW,2BhBuxM2B,QgBvxM3B,CAAX,ChBuxMO,C;O;KAJX,C;4GAOA,yB;MAAA,wD;MgBjyMA,sC;MAAA,oC;MAAA,iCAOe,yB;QAxFf,8D;eAwFe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA/Ed,cAAc,SA+EgB,CA/EhB,CAAd,EAA2B,SA+EM,CA/EN,CAA3B,C;W;S;OA+EI,C;MhB0xmMf,sC;QAI,OAAO,sBgB9xMP,eAAW,2BhB8xmM2B,QgB9xmM3B,CAAX,ChB8xmMO,C;O;KAJX,C;4GAOA,yB;MAAA,wD;MgBxyMA,sC;MAAA,oC;MAAA,iCAOe,yB;QAxFf,8D;eAwFe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA/Ed,cAAc,SA+EgB,CA/EhB,CAAd,EAA2B,SA+EM,CA/EN,CAA3B,C;W;S;OA+EI,C;MhBiyMf,sC;QAI,OAAO,sBgBryMP,eAAW,2BhBqyM2B,QgBryM3B,CAAX,ChBqyMO,C;O;KAJX,C;4GAOA,yB;MAAA,wD;MgB/yMA,sC;MAAA,oC;MAAA,iCAOe,yB;QAxFf,8D;eAwFe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA/Ed,cAAc,SA+EgB,CA/EhB,CAAd,EAA2B,SA+EM,CA/EN,CAA3B,C;W;S;OA+EI,C;MhBwyMf,sC;QAI,OAAO,sBgB5yMP,eAAW,2BhB4yM2B,QgB5yM3B,CAAX,ChB4yMO,C;O;KAJX,C;4GAOA,yB;MAAA,wD;MgBtzMA,sC;MAAA,oC;MAAA,iCAOe,yB;QAxFf,8D;eAwFe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA/Ed,cAAc,SA+EgB,CA/EhB,CAAd,EAA2B,SA+EM,CA/EN,CAA3B,C;W;S;OA+EI,C;MhB+yMf,sC;QAI,OAAO,sBgBnzMP,eAAW,2BhBmzM2B,QgBnzM3B,CAAX,ChBmzMO,C;O;KAJX,C;IAOA,qC;MAMI,OAAO,sBAAW,cAAW,C;K;IAGX,uC;MAIoB,kBel1KQ,iB;Mf1KA,iB;MAAxB,OAAiC,WiBx2M1B,WjBw2M0B,C;K;IAGrC,uC;MAIoB,kBe/0KQ,iB;Mf+0KA,iB;MAAxB,OAAiC,WiB/2M1B,WjB+2M0B,C;K;IAGrC,uC;MAIoB,kBe50KQ,iB;Mf40KA,iB;MAAxB,OAAiC,WiBt3M1B,WjBs3M0B,C;K;IAGrC,uC;MAIoB,kBAAT,oB;MAAiB,mB;MAAxB,OAAiC,WiB73M1B,WjB63M0B,C;K;IAGrC,uC;MAIoB,kBev0KQ,iB;Mfu0KA,iB;MAAxB,OAAiC,WiBp4M1B,WjBo4M0B,C;K;IAGrC,uC;MAIoB,kBep0KQ,iB;Mfo0KA,iB;MAAxB,OAAiC,WiB34M1B,WjB24M0B,C;K;IAGrC,uC;MAIoB,kBAAT,oB;MAAiB,iB;MAAxB,OAAiC,WiB15M1B,WjBk5M0B,C;K;IAGrC,2C;MAMI,OAAmC,OAA5B,2BAAGB,UAAhB,CAA4B,C;K;IAGvC,6C;MAIoB,kBAAf,yB;MAAuB,iC;MAA9B,OAAqD,OiBl6M9C,WjBk6M8C,C;K;IAGzD,6C;MAIoB,kBAAf,yB;MAAuB,iC;MAA9B,OAAqD,OiBz6M9C,WjBy6M8C,C;K;IAGzD,6C;MAIoB,kBAAf,yB;MAAuB,iC;MAA9B,OAAqD,OiBv7M9C,WjBu7M8C,C;K;IAGzD,6C;MAIoB,kBAAf,yB;MAAuB,iC;MAA9B,OAAqD,OiB97M9C,WjB87M8C,C;K;IAGzD,6C;MAIoB,kBAAf,yB;MAAu

B,iC;MAA9B,OAAqD,OiBr8M9C,WjBq8M8C,C;K;IAGzD,6C;MAI0B,kBAAf,yB;MAAuB,iC;MAA9B,OAAqD,  
OiB58M9C,WjB48M8C,C;K;IAGzD,6C;MAI0B,kBAAf,0B;MAAuB,iC;MAA9B,OAAqD,OiBn9M9C,WjBm9M8  
C,C;K;IAkoCrD,gC;MAAQ,oBAAS,CAAT,EAAY,wBAAZ,C;K;IAMR,kC;MAAQ,oBAAS,CAAT,EAAY,0BAA  
Z,C;K;IAMR,kC;MAAQ,oBAAS,CAAT,EAAY,0BAAZ,C;K;IAMR,kC;MAAQ,oBAAS,CAAT,EAAY,0BAAZ,C;  
K;IAMR,kC;MAAQ,oBAAS,CAAT,EAAY,0BAAZ,C;K;IAMR,kC;MAAQ,oBAAS,CAAT,EAAY,0BAAZ,C;K;I  
AMR,kC;MAAQ,oBAAS,CAAT,EAAY,0BAAZ,C;K;IAMR,kC;MAAQ,oBAAS,CAAT,EAAY,0BAAZ,C;K;IAM  
R,kC;MAAQ,oBAAS,CAAT,EAAY,0BAAZ,C;K;IAFAEZ,qB;MAKI,OAAO,qBAAQ,C;K;sFAGnB,qB;MAKI,OA  
AO,qBAAQ,C;K;sFAGnB,qB;MAKI,OAAO,qBAAQ,C;K;sFAGnB,qB;MAKI,OAAO,qBAAQ,C;K;sFAGnB,qB;  
MAKI,OAAO,qBAAQ,C;K;sFAGnB,qB;MAKI,OAAO,qBAAQ,C;K;sFAGnB,qB;MAKI,OAAO,qBAAQ,C;K;sF  
AGnB,qB;MAKI,OAAO,qBAAQ,C;K;sFAGnB,qB;MAKI,OAAO,qBAAQ,C;K;0FAGnB,qB;MAKI,OAAO,EAxE  
A,qBAAQ,CAwER,C;K;4FAGX,qB;MAKI,OAAO,EAxEA,qBAAQ,CAwER,C;K;4FAGX,qB;MAKI,OAAO,EAx  
EA,qBAAQ,CAwER,C;K;4FAGX,qB;MAKI,OAAO,EAxEA,qBAAQ,CAwER,C;K;4FAGX,qB;MAKI,OAAO,EA  
xEA,qBAAQ,CAwER,C;K;4FAGX,qB;MAKI,OAAO,EAxEA,qBAAQ,CAwER,C;K;4FAGX,qB;MAKI,OAAO,EA  
xEA,qBAAQ,CAwER,C;K;4FAGX,qB;MAKI,OAAO,EAxEA,qBAAQ,CAwER,C;K;4FAGX,qB;MAKI,OAAO,EA  
xEA,qBAAQ,CAwER,C;K;IAOP,kC;MAAQ,0BAAO,CAAP,I;K;IAMR,oC;MAAQ,0BAAO,CAAP,I;K;IAMR,  
oC;MAAQ,0BAAO,CAAP,I;K;IAMR,oC;MAAQ,0BAAO,CAAP,I;K;IAMR,oC;MAAQ,0BAAO,CAAP,I;K;IAM  
R,oC;MAAQ,0BAAO,CAAP,I;K;IAMR,oC;MAAQ,0BAAO,CAAP,I;K;IAMR,oC;MAAQ,0BAAO,CAAP,I;K;IA  
MR,oC;MAAQ,0BAAO,CAAP,I;K;IA8TZ,yD;MAcI,sBAAS,cAAT,EAAYB,SAAzB,EAAoC,OAApC,C;K;IAGJ,y  
D;MAYI,mBAAK,SAAL,EAAGB,OAAhB,C;MACA,qBAAQ,SAAR,EAAMB,OAAAnB,C;K;IAGJ,yD;MAYI,mBA  
AK,SAAL,EAAGB,OAAhB,C;MACA,sBAAQ,SAAR,EAAMB,OAAAnB,C;K;IAGJ,0D;MAYI,mBAAK,SAAL,EA  
AGB,OAAhB,C;MACA,sBAAQ,SAAR,EAAMB,OAAAnB,C;K;IAGJ,0D;MAYI,mBAAK,SAAL,EAAGB,OAAhB,  
C;MACA,sBAAQ,SAAR,EAAMB,OAAAnB,C;K;IAGJ,0D;MAYI,mBAAK,SAAL,EAAGB,OAAhB,C;MACA,sBA  
AQ,SAAR,EAAMB,OAAAnB,C;K;IAGJ,0D;MAYI,mBAAK,SAAL,EAAGB,OAAhB,C;MACA,sBAAQ,SAAR,EA  
AMB,OAAAnB,C;K;IAGJ,0D;MAYI,mBAAK,SAAL,EAAGB,OAAhB,C;MACA,sBAAQ,SAAR,EAAMB,OAAAnB,  
C;K;IA2B0B,oD;MAAA,wB;QAAW,2BAAK,KAAL,C;O;K;IAJzC,mC;MAII,OAAO,qBAAa,gBAAb,EAAMB,gC  
AAAnB,C;K;IAOgB,8C;MAAA,wB;QAAW,wBAAK,KAAL,C;O;K;IAJtC,gC;MAII,OAAO,+BAAU,gBAAV,GAA  
gB,6BAAhB,C;K;IAOgB,8C;MAAA,wB;QAAW,wBAAK,KAAL,C;O;K;IAJtC,gC;MAII,OAAO,kBAAU,gBAA  
V,EAAGB,6BAAhB,C;K;IAOkB,kD;MAAA,wB;QAAW,0BAAK,KAAL,C;O;K;IAJxC,kC;MAII,OAAO,kCAAY,  
gBAAZ,GAAkB,+BAAIB,C;K;IAOiB,gD;MAAA,wB;QAAW,yBAAK,KAAL,C;O;K;IAJvC,iC;MAII,OAAO,kC  
AAW,gBAAX,GAAiB,8BAAjB,C;K;IAOe,4C;MAAA,wB;QAAW,uBAAK,KAAL,C;O;K;IAJrC,+B;MAII,OAAO  
,gCAAS,gBAAT,GAAe,4BAAf,C;K;IAOgB,8C;MAAA,wB;QAAW,wBAAK,KAAL,C;O;K;IAJtC,gC;MAII,OAA  
O,kBAAU,gBAAV,EAAGB,6BAAhB,C;K;IAOiB,gD;MAAA,wB;QAAW,yBAAK,KAAL,C;O;K;IAJvC,iC;MAII,  
OAAO,gCAAW,gBAAX,GAAiB,8BAAjB,C;K;wFA2CX,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,uC;QAW  
LeAAiC,cAAIB,YAAY,gBAAZ,CAAKB,EAAC,EAAD,C;QAC1B,kBAAY,mBAAoB,QAApB,C;QAYqBH,Q;QAA  
hB,iD;UAAgB,cAAhB,e;UACI,WA1qB8C,SA0qB/B,CAAU,OAAV,C;UOX+QnB,wBAAI,IAAK,MAAT,EAAGB,  
IAAK,OAArB,C;;QP8zPA,OA4qBO,W;O;KAXrBX,C;0FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,uC;Q  
AWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAKB,EAAC,EAAD,C;QAC1B,kBAAY,mBAAoB,QAApB,C;QAYqBH,Q;  
QAAhB,iD;UAAgB,cAAhB,e;UACI,WA1qB8C,SA0qB/B,CAAU,OAAV,C;UOV/QnB,wBAAI,IAAK,MAAT,EA  
AGB,IAAK,OAArB,C;;QP60PA,OA4qBO,W;O;KAXrBX,C;0FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,  
uC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAKB,EAAC,EAAD,C;QAC1B,kBAAY,mBAAoB,QAApB,C;QAYqB  
H,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WA1qB8C,SA0qB/B,CAAU,OAAV,C;UOTgRnB,wBAAI,IAAK,MAAT  
,EAAGB,IAAK,OAArB,C;;QP41PA,OA4qBO,W;O;KAXrBX,C;0FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MA  
AA,uC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAKB,EAAC,EAAD,C;QAC1B,kBAAY,mBAAoB,QAApB,C;QA  
yqBH,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WA1qB8C,SA0qB/B,CAAU,OAAV,C;UORhRnB,wBAAI,IAAK,M  
AAT,EAAGB,IAAK,OAArB,C;;QP22PA,OA4qBO,W;O;KAXrBX,C;0FAeA,yB;MAAA,0D;MAAA,yD;MAAA,u  
E;MAAA,uC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAKB,EAAC,EAAD,C;QAC1B,kBAAY,mBAAoB,QAApB,  
C;QAYqBH,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WA1qB8C,SA0qB/B,CAAU,OAAV,C;UOPiRnB,wBAAI,IAA  
K,MAAT,EAAGB,IAAK,OAArB,C;;QP03PA,OA4qBO,W;O;KAXrBX,C;0FAeA,yB;MAAA,0D;MAAA,yD;MAA

A,uE;MAAA,uC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAY,mBAAoB,QAApB,C;QAYqBH,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WA1qB8C,SA0qB/B,CAAU,OAAV,C;UOnjRnB,wBAAI,IAAK,MAAT,EAAGB,IAAK,OAArB,C;;QPpy4PA,OA4qBO,W;O;KAxrBX,C;0FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,uC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAY,mBAAoB,QAApB,C;QAYqBH,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WA1qB8C,SA0qB/B,CAAU,OAAV,C;UOlkRnB,wBAAI,IAAK,MAAT,EAAGB,IAAK,OAArB,C;;QPw5PA,OA4qBO,W;O;KAxrBX,C;0FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,uC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAY,mBAAoB,QAApB,C;QAYqBH,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WA1qB8C,SA0qB/B,CAAU,OAAV,C;UOjlRnB,wBAAI,IAAK,MAAT,EAAGB,IAAK,OAArB,C;;QPu6PA,OA4qBO,W;O;KAxrBX,C;0FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAA4qBA,oC;MAAA,gC;MA5qBA,uC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAY,mBAAoB,QAApB,C;QAYqBH,Q;QAAhB,iD;UAAgB,cAAhB,0B;UACI,WA1qB8C,SA0qB/B,CAAU,oBAAV,C;UOhmRnB,wBAAI,IAAK,MAAT,EAAGB,IAAK,OAArB,C;;QP7PA,OA4qBO,W;O;KAxrBX,C;4FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAAoB,QAApB,C;QAmQL,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WAAY,aApQoC,WAOqHC,CAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;QApQhB,OASQO,W;O;KAIRX,C;8FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAAwB,QAAxB,C;QAqQL,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WAAY,aArQuC,WAqQnC,CAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;QArQhB,OAuQO,W;O;KANRX,C;8FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAAwB,QAAxB,C;QAqQL,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WAAY,aAtQwC,WASqPC,CAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;QAtQhB,OAwQO,W;O;KApRX,C;8FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAAwB,QAAxB,C;QASQL,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WAAY,aAvQsC,WAuQIC,CAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;QAvQhB,OAYQO,W;O;KARRX,C;8FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAAuB,QAAvB,C;QAuQL,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WAAY,aAxQuC,WAwQnC,CAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;QAxQhB,OA0QO,W;O;KATRX,C;8FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAAwB,QAAxB,C;QAwQL,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WAAY,aAZQwC,WAYqPC,CAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;QAZqHb,OA2QO,W;O;KAvRX,C;8FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAAyB,QAAzB,C;QAYQL,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WAAY,aA1QyC,WA0QrC,CAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;QA1QhB,OA4QO,W;O;KAxRX,C;8FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAA0B,QAA1B,C;QA0QL,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WAAY,aA3Q0C,WA2QIC,CAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;QA3QhB,OA6QO,W;O;KAZRX,C;8FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAA6QA,oC;MAAA,gC;MA7QA,yC;QAWI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAAuB,QAAvB,C;QA2QL,Q;QAAhB,iD;UAAgB,cAAhB,0B;UACI,WAAY,aA5QuC,WA4QnC,CAAY,oBAAZ,CAAJ,EAA0B,oBAA1B,C;;QA5QhB,OA8QO,W;O;KA1RX,C;8FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yD;QAUI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAAoB,QAApB,C;QA6QL,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WAAY,aA9QoC,WA8QhC,CAAY,OAAZ,CAAJ,EA9QiD,cA8QvB,CAAe,OAaf,CAA1B,C;;QA9QhB,OAgrO,W;O;KA3RX,C;8FACa,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yD;QAUI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAAoB,QAApB,C;QA+QL,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WAAY,aAhRoC,WAgRhC,CAAY,OAAZ,CAAJ,EAhRiD,cAgRvB,CAAe,OAaf,CAA1B,C;;QAhRhB,OAkRO,W;O;KA7RX,C;+FACa,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yD;QAUI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAAoB,QAApB,C;QAiRL,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WAAY,aAIrOC,WakRhC,CAAY,OAAZ,CAAJ,EAIrID,cAkRvB,CAAe,OAaf,CAA1B,C;;QAIRhB,OAoRO,W;O;KA/RX,C;+FACa,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yD;QAUI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAC,EAAd,C;QAC1B,kBAAc,mBAAoB,QAApB,C;QAmRL,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,WAAY,aApRoC,WAOrhC,

CAAY,OAAZ,CAAJ,EApRiD,cAoRvB,CAAe,OAAf,CAA1B,C;;QApRhB,OAsRO,W;O;KAjSX,C;+FAcA,yB;M  
AAA,0D;MAAA,yD;MAAA,uE;MAAA,yD;QAUI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAc,EAAc,C;QAC1  
B,kBAAc,mBAAoB,QAAPB,C;QAqRL,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,WAAY,aAtRoC,WAsRhC,CAAY  
,OAAZ,CAAJ,EA1RiD,cAsRvB,CAAe,OAAf,CAA1B,C;;QAtRhB,OAwRO,W;O;KAnSX,C;+FAcA,yB;MAAA,0  
D;MAAA,yD;MAAA,uE;MAAA,yD;QAUI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAc,EAAc,C;QAC1B,kBA  
Ac,mBAAoB,QAAPB,C;QAuRL,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,WAAY,aAxRoC,WAwRhC,CAAY,OAA  
Z,CAAJ,EAxRiD,cAwRvB,CAAe,OAAf,CAA1B,C;;QAxRhB,OA0RO,W;O;KArSX,C;+FAcA,yB;MAAA,0D;M  
AAA,yD;MAAA,uE;MAAA,yD;QAUI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAc,EAAc,C;QAC1B,kBAAc,m  
BAAoB,QAAPB,C;QAyRL,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,WAAY,aA1RoC,WA0RhC,CAAY,OAAZ,CA  
AJ,EA1RiD,cA0RvB,CAAe,OAAf,CAA1B,C;;QA1RhB,OA4RO,W;O;KAvSX,C;+FAcA,yB;MAAA,0D;MAAA,y  
D;MAAA,uE;MAAA,yD;QAUI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAc,EAAc,C;QAC1B,kBAAc,mBAAo  
B,QAAPB,C;QA2RL,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,WAAY,aA5RoC,WA4RhC,CAAY,OAAZ,CAAJ,EA  
5RiD,cA4RvB,CAAe,OAAf,CAA1B,C;;QA5RhB,OA8RO,W;O;KAZSX,C;+FAcA,yB;MAAA,0D;MAAA,yD;MA  
AA,uE;MA8RA,oC;MAAA,gC;MA9RA,yD;QAUI,eAAiC,cAAIB,YAAY,gBAAZ,CAAkB,EAAc,EAAc,C;QAC1  
B,kBAAc,mBAAoB,QAAPB,C;QA6RL,Q;QAaHb,iD;UAAgB,cAAhB,0B;UACI,WAAY,aA9RoC,WA8RhC,CA  
AY,oBAAZ,CAAJ,EA9RiD,cA8RvB,CAAe,oBAAf,CAA1B,C;;QA9RhB,OAgsO,W;O;KA3SX,C;gGAcA,+C;MA  
UoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAY,aAAI,YAAY,OAAZ,CAAJ,EAA  
0B,OAA1B,C;;MAEhB,OAAO,W;K;kGAGX,+C;MAUoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SA  
AhB,M;QACI,WAAY,aAAI,YAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;MAEhB,OAAO,W;K;kGAGX,+C;MAUoB,  
Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAY,aAAI,YAAY,OAAZ,CAAJ,EAA0B,O  
AA1B,C;;MAEhB,OAAO,W;K;iGAGX,+C;MAUoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;  
QACI,WAAY,aAAI,YAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;MAEhB,OAAO,W;K;kGAGX,+C;MAUoB,Q;MA  
AhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAY,aAAI,YAAY,OAAZ,CAAJ,EAA0B,OAA1B,  
C;;MAEhB,OAAO,W;K;kGAGX,+C;MAUoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,  
WAAY,aAAI,YAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;MAEhB,OAAO,W;K;kGAGX,+C;MAUoB,Q;MAAhB,w  
BAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAY,aAAI,YAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;MA  
EhB,OAAO,W;K;kGAGX,+C;MAUoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAA  
Y,aAAI,YAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;MAEhB,OAAO,W;K;iGAGX,yB;MAAA,oC;MAAA,gC;MAA  
A,sD;QAUoB,Q;QAaHb,wBAAGB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UACI,WAAY,aAAI,YAAY,o  
BAAZ,CAAJ,EAA0B,oBAA1B,C;;QAEhB,OAAO,W;O;KAbX,C;kGAGBA,+D;MAUoB,Q;MAAhB,wBAAGB,S  
AAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAY,aAAI,YAAY,OAAZ,CAAJ,EAA0B,eAAe,OAAf,CAA1B,C;;  
MAEhB,OAAO,W;K;kGAGX,+D;MAUoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,W  
AAY,aAAI,YAAY,OAAZ,CAAJ,EAA0B,eAAe,OAAf,CAA1B,C;;MAEhB,OAAO,W;K;mGAGX,+D;MAUoB,Q;  
MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAY,aAAI,YAAY,OAAZ,CAAJ,EAA0B,eAA  
e,OAAf,CAA1B,C;;MAEhB,OAAO,W;K;mGAGX,+D;MAUoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,  
SAAhB,M;QACI,WAAY,aAAI,YAAY,OAAZ,CAAJ,EAA0B,eAAe,OAAf,CAA1B,C;;MAEhB,OAAO,W;K;mGA  
GX,+D;MAUoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAY,aAAI,YAAY,OAAZ,  
CAAJ,EAA0B,eAAe,OAAf,CAA1B,C;;MAEhB,OAAO,W;K;mGAGX,+D;MAUoB,Q;MAAhB,wBAAGB,SAAhB  
,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAY,aAAI,YAAY,OAAZ,CAAJ,EAA0B,eAAe,OAAf,CAA1B,C;;MAEh  
B,OAAO,W;K;mGAGX,+D;MAUoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAY,  
aAAI,YAAY,OAAZ,CAAJ,EAA0B,eAAe,OAAf,CAA1B,C;;MAEhB,OAAO,W;K;mGAGX,+D;MAUoB,Q;MAA  
hB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAY,aAAI,YAAY,OAAZ,CAAJ,EAA0B,eAAe,OA  
Af,CAA1B,C;;MAEhB,OAAO,W;K;mGAGX,yB;MAAA,oC;MAAA,gC;MAAA,sE;QAUoB,Q;QAaHb,wBAAGB  
,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UACI,WAAY,aAAI,YAAY,oBAAZ,CAAJ,EAA0B,eAAe,oBAA  
f,CAA1B,C;;QAEhB,OAAO,W;O;KAbX,C;2FAGBA,6C;MASoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAA  
A,SAAhB,M;QACI,WAAe,UAAU,OAAV,C;QOx+QnB,wBAAL,IAAK,MAAT,EAAgB,IAAK,OAARb,C;;MP0+Q  
A,OAAO,W;K;8FAGX,6C;MASoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAe,U  
AAU,OAAV,C;QOv/QnB,wBAAL,IAAK,MAAT,EAAgB,IAAK,OAARb,C;;MPy/QA,OAAO,W;K;8FAGX,6C;M

ASoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAe,UAAU,OAAV,C;QOtgRnB,wB  
AAI,IAAK,MAAT,EAAGB,IAAK,OAArB,C;;MPwgRA,OAAO,W;K;8FAGX,6C;MASoB,Q;MAAhB,wBAAgB,S  
AAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAe,UAAU,OAAV,C;QOrhRnB,wBAAI,IAAK,MAAT,EAAGB,IA  
AK,OAArB,C;;MPuhRA,OAAO,W;K;8FAGX,6C;MASoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SA  
AhB,M;QACI,WAAe,UAAU,OAAV,C;QOpiRnB,wBAAI,IAAK,MAAT,EAAGB,IAAK,OAArB,C;;MPsiRA,OAAO  
,W;K;8FAGX,6C;MASoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAe,UAAU,OA  
AV,C;QOnjRnB,wBAAI,IAAK,MAAT,EAAGB,IAAK,OAArB,C;;MPqjRA,OAAO,W;K;8FAGX,6C;MASoB,Q;M  
AAhB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAe,UAAU,OAAV,C;QOlkRnB,wBAAI,IAAK,  
MAAT,EAAGB,IAAK,OAArB,C;;MPokRA,OAAO,W;K;8FAGX,6C;MASoB,Q;MAAhB,wBAAgB,SAAhB,gB;Q  
AAGB,cAAA,SAAhB,M;QACI,WAAe,UAAU,OAAV,C;QOjlRnB,wBAAI,IAAK,MAAT,EAAGB,IAAK,OAArB,  
C;;MPmlRA,OAAO,W;K;8FAGX,yB;MAAA,oC;MAAA,gC;MAAA,oD;QASoB,Q;QAAhB,wBAAgB,SAAhB,gB  
;UAAgB,cAAhB,UAAgB,SAAhB,O;UACI,WAAe,UAAU,oBAAV,C;UOhmRnB,wBAAI,IAAK,MAAT,EAAGB,I  
AAK,OAArB,C;;QPkmRA,OAAO,W;O;KAZX,C;gGAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,2C;QAY  
I,aAAa,mBAAsC,cAAIB,YAAY,gBAAZ,CAAKB,EAAC,EAAD,CAATC,C;QAsJG,Q;QAAhB,iD;UAAgB,cAAhB,e  
;UArJuB,MAsJP,aAAI,OAAJ,EATJe,aAsJF,CAAC,OAAD,CAAB,C;;QAtJhB,OAAuB,M;O;KAd3B,C;kGAgBA,yB;  
MAAA,0D;MAAA,yD;MAAA,uE;MAAA,2C;QAaI,aAAa,mBAAyC,cAAIB,YAAY,gBAAZ,CAAKB,EAAC,EA  
AD,CAAzC,C;QAsJG,Q;QAAhB,iD;UAAgB,cAAhB,e;UArJuB,MAsJP,aAAI,OAAJ,EATJe,aAsJF,CAAC,OA  
AD,CAAB,C;;QAtJhB,OAAuB,M;O;KAd3B,C;kGaiBA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,2C;QAaI,aAAa,m  
BAA0C,cAAIB,YAAY,gBAAZ,CAAKB,EAAC,EAAD,CAA1C,C;QAsJG,Q;QAAhB,iD;UAAgB,cAAhB,e;UArJuB  
,MAsJP,aAAI,OAAJ,EATJe,aAsJF,CAAC,OAAD,CAAB,C;;QAtJhB,OAAuB,M;O;KAd3B,C;kGaiBA,yB;MAAA,0  
D;MAAA,yD;MAAA,uE;MAAA,2C;QAaI,aAAa,mBAAwC,cAAIB,YAAY,gBAAZ,CAAKB,EAAC,EAAD,CAAx  
C,C;QAsJG,Q;QAAhB,iD;UAAgB,cAAhB,e;UArJuB,MAsJP,aAAI,OAAJ,EATJe,aAsJF,CAAC,OAAD,CAAB,C;;Q  
AtJhB,OAAuB,M;O;KAd3B,C;kGaiBA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,2C;QAaI,aAAa,mBAAyC,  
cAAIB,YAAY,gBAAZ,CAAKB,EAAC,EAAD,CAAzC,C;QAsJG,Q;QAAhB,iD;UAAgB,cAAhB,e;UArJuB,MAsJP,  
aAAI,OAAJ,EATJe,aAsJF,CAAC,OAAD,CAAB,C;;QAtJhB,OAAuB,M;O;KAd3B,C;kGaiBA,yB;MAAA,0D;MAA  
A,yD;MAAA,uE;MAAA,2C;QAaI,aAAa,mBAA0C,cAAIB,YAAY,gBAAZ,CAAKB,EAAC,EAAD,CAA1C,C;QAsJ  
G,Q;QAAhB,iD;UAAgB,cAAhB,e;UArJuB,MAsJP,aAAI,OAAJ,EATJe,aAsJF,CAAC,OAAD,CAAB,C;;QAtJhB,OA  
AuB,M;O;KAd3B,C;kGaiBA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,2C;QAaI,aAAa,mBAA2C,cAAIB,Y  
AAY,gBAAZ,CAAKB,EAAC,EAAD,CAA3C,C;QAsJG,Q;QAAhB,iD;UAAgB,cAAhB,e;UArJuB,MAsJP,aAAI,O  
AAJ,EATJe,aAsJF,CAAC,OAAD,CAAB,C;;QAtJhB,OAAuB,M;O;KAd3B,C;kGaiBA,yB;MAAA,0D;MAAA,yD;M  
AAA,uE;MAAA,2C;QAaI,aAAa,mBAA4C,cAAIB,YAAY,gBAAZ,CAAKB,EAAC,EAAD,CAA5C,C;QAsJG,Q;Q  
AAhB,iD;UAAgB,cAAhB,e;UArJuB,MAsJP,aAAI,OAAJ,EATJe,aAsJF,CAAC,OAAD,CAAB,C;;QAtJhB,OA  
AuB,M;O;KAd3B,C;kGaiBA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAwJA,oC;MAAA,gC;MAxJA,2C;Q  
AaI,aAAa,mBAA2D,cAApC,YAAiB,aAAL,gBAAK,EAAa,GAAb,CAAjB,CAAoC,EAAC,EAAD,CAA3D,C;QAsJ  
G,Q;QAAhB,iD;UAAgB,cAAhB,0B;UArJuB,MAsJP,aAAI,oBAAJ,EATJe,aAsJF,CAAC,oBAAAd,CAAB,C;;QAtJhB  
,OAAuB,M;O;KAd3B,C;oGaiBA,iD;MAUoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,  
WAAy,aAAI,OAAJ,EAAa,cAAc,OAAD,CAAB,C;;MAEhB,OAAO,W;K;sGAGX,iD;MAWoB,Q;MAAhB,wBAA  
gB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAy,aAAI,OAAJ,EAAa,cAAc,OAAD,CAAB,C;;MAEhB,OAA  
O,W;K;sGAGX,iD;MAWoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAy,aAAI,O  
AAJ,EAAa,cAAc,OAAD,CAAB,C;;MAEhB,OAAO,W;K;sGAGX,iD;MAWoB,Q;MAAhB,wBAAgB,SAAhB,gB;Q  
AAGB,cAAA,SAAhB,M;QACI,WAAy,aAAI,OAAJ,EAAa,cAAc,OAAD,CAAB,C;;MAEhB,OAAO,W;K;sGAGX,i  
D;MAWoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAy,aAAI,OAAJ,EAAa,cAAc,  
OAAD,CAAB,C;;MAEhB,OAAO,W;K;sGAGX,iD;MAWoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SA  
AhB,M;QACI,WAAy,aAAI,OAAJ,EAAa,cAAc,OAAD,CAAB,C;;MAEhB,OAAO,W;K;sGAGX,iD;MAWoB,Q;M  
AAhB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,WAAy,aAAI,OAAJ,EAAa,cAAc,OAAD,CAAB,C;;  
MAEhB,OAAO,W;K;sGAGX,iD;MAWoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,W  
AAY,aAAI,OAAJ,EAAa,cAAc,OAAD,CAAB,C;;MAEhB,OAAO,W;K;sGAGX,yB;MAAA,oC;MAAA,gC;MAAA,  
wD;QAWoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UACI,WAAy,aAAI,oBAAJ,E



AAN,I;MAAA,QAAM,gBAAN,C;aACH,C;UAAK,iB;UAAL,K;aACA,C;UAAK,aAAM,UAAK,CAAL,CAAN,C;UAAL,K;gBACQ,iCAAA,qBAAqB,YAAY,gBAAZ,CAArB,CAAb,C;UAHL,K;;MAAP,W;K;IAOJ,4B;MAMiB,IAAN,I;MAAA,QAAM,gBAAN,C;aACH,C;UAAK,iB;UAAL,K;aACA,C;UAAK,aAAM,UAAK,CAAL,CAAN,C;UAAL,K;gBACQ,iCAAA,qBAAMb,YAAY,gBAAZ,CAAnB,CAAb,C;UAHL,K;;MAAP,W;K;IAOJ,4B;MAMiB,IAAN,I;MAAA,QAAM,gBAAN,C;aACH,C;UAAK,iB;UAAL,K;aACA,C;UAAK,aAAM,UAAK,CAAL,CAAN,C;UAAL,K;gBACQ,iCAAA,qBAAoB,YAAY,gBAAZ,CAApB,CAAb,C;UAHL,K;;MAAP,W;K;IAOJ,4B;MAMiB,IAAN,I;MAAA,QAAM,gBAAN,C;aACH,C;UAAK,iB;UAAL,K;aACA,C;UAAK,aAAM,UAAK,CAAL,CAAN,C;UAAL,K;gBACQ,iCAAA,qBAAqB,YAAY,gBAAZ,CAArB,CAAb,C;UAHL,K;;MAAP,W;K;IAOJ,4B;MAMiB,IAAN,I;MAAA,QAAM,gBAAN,C;aACH,C;UAAK,iB;UAAL,K;aACA,C;UAAK,aAAM,UAAK,CAAL,CAAN,C;UAAL,K;gBACQ,iCAAA,qBAAsB,YAAY,gBAAZ,CAAtB,CAAb,C;UAHL,K;;MAAP,W;K;IAOJ,4B;MAMiB,IAAN,I;MAAA,QAAM,gBAAN,C;aACH,C;UAAK,iB;UAAL,K;aACA,C;UAAK,aAAM,UAAK,CAAL,CAAN,C;UAAAL,K;gBACQ,iCAAA,qBAAuB,YAAY,gBAAZ,CAAvB,CAAb,C;UAHL,K;;MAAP,W;K;IAOJ,4B;MAMiB,IAAN,I;MAAA,QAAM,gBAAN,C;aACH,C;UAAK,iB;UAAL,K;aACA,C;UAAK,aAAM,sBAAK,CAAL,EAAN,C;UAAAL,K;gBACQ,iCAAA,qBAAoB,YAAiB,eAAL,gBAAK,EAAa,GAAb,CAAjB,CAApB,CAAb,C;UAHL,K;;MAAP,W;K;oFAOJ,yB;MAAA,+D;MAwaA,gD;MAxaA,uC;QAMW,kBAAU,gB;QAsaD,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,WaVa6B,SAualB,CAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAxahB,OA0aO,W;O;KAhX,C;sFASA,yB;MAAA,+D;MA0aA,gD;MA1aA,uC;QAMW,kBAAU,gB;QAwaD,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,WaZa6B,SAyalB,CAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QA1ahB,OA4aO,W;O;KAlbX,C;sFASA,yB;MAAA,+D;MA4aA,gD;MA5aA,uC;QAMW,kBAAU,gB;QA0aD,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,WA3a6B,SA2alB,CAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QA5ahB,OA8aO,W;O;KApbX,C;sFASA,yB;MAAA,+D;MA8aA,gD;MA9aA,uC;QAMW,kBAAU,gB;QA4aD,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,WA7a6B,SA6alB,CAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QA9ahB,OAgbO,W;O;KAtbX,C;sFASA,yB;MAAA,+D;MAGbA,gD;MAhbA,uC;QAMW,kBAAU,gB;QA8aD,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,Wa/a6B,SA+alB,CAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAhhbB,OAkbO,W;O;KAxbX,C;sFASA,yB;MAAA,+D;MAkbA,gD;MAlbA,uC;QAMW,kBAAU,gB;QAgbD,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,Wajb6B,SAibLB,CAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QA1bhB,OAobO,W;O;KA1bX,C;sFASA,yB;MAAA,+D;MAobA,gD;MApbA,uC;QAMW,kBAAU,gB;QAkbD,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,WAnb6B,SAmbLB,CAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QApbhB,OAsoB,W;O;KA5bX,C;sFASA,yB;MAAA,+D;MAsbA,gD;MAtbA,uC;QAMW,kBAAU,gB;QAobD,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,WArb6B,SAqblB,CAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAtbhB,OAwbO,W;O;KA9bX,C;sFASA,yB;MAAA,+D;MAwbA,oC;MAAA,gD;MAAA,gC;MAxbA,uC;QAMW,kBAAU,gB;QAsbD,Q;QAaHb,iD;UAAgB,cAAhB,0B;UACI,Wavb6B,SAublB,CAAU,oBAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAxhbB,OA0bO,W;O;KAhcX,C;sFASA,yB;MAAA,+D;MA0bA,gD;MA1bA,uC;QAUW,kBAAU,gB;QAwbD,Q;QAaHb,iD;UAAgB,cAAhB,e;UACI,Wazb6B,SAyblB,CAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QA1bhB,OA4bO,W;O;KAtcX,C;KgAaA,yB;MAAA,+D;MAsjA,gD;MATjA,uC;QAYW,kBAAiB,gB;QAqJR,gB;QADhB,YAAY,C;QACZ,iD;UAAgB,cAAhB,e;UACI,WAtJoC,SAsjzB,EAAU,cAAV,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAvJhB,OAyJO,W;O;KArKX,C;oGAeA,yB;MAAA,+D;MAyJA,gD;MAzJA,uC;QAYW,kBAAiB,gB;QAwJR,gB;QADhB,YAAY,C;QACZ,iD;UAAgB,cAAhB,e;UACI,WazJoC,SAyJzB,EAAU,cAAV,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QA1JhB,OA4JO,W;O;KAxKX,C;oGAeA,yB;MAAA,+D;MA4JA,gD;MA5JA,uC;QAYW,kBAAiB,gB;QA2JR,gB;QADhB,YAAY,C;QACZ,iD;UAAgB,cAAhB,e;UACI,WA5JoC,SA4JzB,EAAU,cAAV,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QA7JhB,OA+JO,W;O;KA3KX,C;oGAeA,yB;MAAA,+D;MA+JA,gD;MA/JA,uC;QAYW,kBAAiB,gB;QA8JR,gB;QADhB,YAAY,C;QACZ,iD;UAAgB,cAAhB,e;UACI,WA/JoC,SA+JzB,EAAU,cAAV,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAhhB,OAkKO,W;O;KA9KX,C;oGAeA,yB;MAAA,+D;MAkKA,gD;MAiKA,uC;QAYW,kBAAiB,gB;QAiKR,gB;QADhB,YAAY,C;QACZ,iD;UAAgB,cAAhB,e;UACI,WAIKoC,SakKzB,EAAU,cAAV,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAnKhB,OAqKO,W;O;KAjLX,C;oGAeA,yB;MAAA,+D;MAqKA,gD;MArKA,uC;QAYW,kBAAiB,gB;QAoKR,gB;QADhB,YAAY,C;QACZ,iD;UAAgB,cAAhB,e;UACI,WArKoC,SAqKzB,EAAU,cAAV,EAAU,sBAAV,

WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAtKhB,OAwKO,W;O;KApLX,C;oGAeA,yB;MAA  
A,+D;MAwKA,gD;MAxKA,uC;QAYW,kBAAiB,gB;QAUKR,gB;QADhB,YAAY,C;QACZ,iD;UAAgB,cAAhB,e;  
UACI,WAxKoC,SAwKzB,EAAU,cAAV,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAA  
P,C;;QAZKhB,OA2KO,W;O;KAvLX,C;oGAeA,yB;MAAA,+D;MA2KA,gD;MA3KA,uC;QAYW,kBAAiB,gB;QA  
0KR,gB;QADhB,YAAY,C;QACZ,iD;UAAgB,cAAhB,e;UACI,WA3KoC,SA2KzB,EAAU,cAAV,EAAU,sBAAV,  
WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QA5KhB,OA8KO,W;O;KAILX,C;oGAeA,yB;MAA  
A,+D;MA8KA,oC;MAAA,gD;MAAA,gC;MA9KA,uC;QAYW,kBAAiB,gB;QA6KR,gB;QADhB,YAAY,C;QACZ  
,iD;UAAgB,cAAhB,0B;UACI,WA9KoC,SA8KzB,EAAU,cAAV,EAAU,sBAAV,WAAmB,oBAAnB,C;UACC,OA  
AZ,WAAY,EAAO,IAAP,C;;QA/KhB,OAILO,W;O;KA7LX,C;oGAeA,yB;MAAA,+D;MAiLA,gD;MAjLA,uC;QA  
YW,kBAAiB,gB;QAgLR,gB;QADhB,YAAY,C;QACZ,iD;UAAgB,cAAhB,e;UACI,WajLoC,SaiLzB,EAAU,cAA  
V,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAILhB,OAoLO,W;O;KAhMX,C;  
sGAeA,yB;MAAA,gD;MAAA,oD;QAWoB,UACS,M;QAFzB,YAAY,C;QACZ,wBAAgB,SAAhB,gB;UAAgB,cA  
AA,SAAhB,M;UACI,WAAW,WAAU,cAAV,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,I  
AAP,C;;QAEhB,OAAO,W;O;KafX,C;uGakBA,yB;MAAA,gD;MAAA,oD;QAWoB,UACS,M;QAFzB,YAAY,C;  
QACZ,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,WAAW,WAAU,cAAV,EAAU,sBAAV,WAAmB,  
OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KafX,C;wGakBA,yB;MAAA,gD;MAA  
A,oD;QAWoB,UACS,M;QAFzB,YAAY,C;QACZ,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,WAA  
W,WAAU,cAAV,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;  
O;KafX,C;wGakBA,yB;MAAA,gD;MAAA,oD;QAWoB,UACS,M;QAFzB,YAAY,C;QACZ,wBAAgB,SAAhB,g  
B;UAAgB,cAAA,SAAhB,M;UACI,WAAW,WAAU,cAAV,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,W  
AAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KafX,C;wGakBA,yB;MAAA,gD;MAAA,oD;QAWoB,UACS,M;QA  
FzB,YAAY,C;QACZ,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,WAAW,WAAU,cAAV,EAAU,sBA  
AV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KafX,C;wGakBA,yB;MA  
AA,gD;MAAA,oD;QAWoB,UACS,M;QAFzB,YAAY,C;QACZ,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;  
UACI,WAAW,WAAU,cAAV,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEh  
B,OAAO,W;O;KafX,C;wGakBA,yB;MAAA,gD;MAAA,oD;QAWoB,UACS,M;QAFzB,YAAY,C;QACZ,wBAA  
gB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,WAAW,WAAU,cAAV,EAAU,sBAAV,WAAmB,OAAnB,C;UA  
CC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KafX,C;wGakBA,yB;MAAA,gD;MAAA,oD;QAWoB,  
UACS,M;QAFzB,YAAY,C;QACZ,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,WAAW,WAAU,cAA  
V,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KafX,C;wG  
AkBA,yB;MAAA,oC;MAAA,gD;MAAA,gC;MAAA,oD;QAWoB,UACS,M;QAFzB,YAAY,C;QACZ,wBAAgB,S  
AAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UACI,WAAW,WAAU,cAAV,EAAU,sBAAV,WAAmB,oBAAnB,  
C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KafX,C;wGakBA,yB;MAAA,gD;MAAA,oD;QA  
WoB,UACS,M;QAFzB,YAAY,C;QACZ,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,WAAW,WAAU,  
cAAV,EAAU,sBAAV,WAAmB,OAAnB,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KafX,C;  
uFakBA,yB;MAAA,gD;MAAA,oD;QAIoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,W  
AAW,UAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KARX,C;0FAWA,yB;MAA  
A,gD;MAAA,oD;QAIoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,WAAW,UAAU,OAA  
V,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KARX,C;0FAWA,yB;MAAA,gD;MAAA,oD;Q  
AIoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,WAAW,UAAU,OAAV,C;UACC,OAAZ,  
WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KARX,C;0FAWA,yB;MAAA,gD;MAAA,oD;QAIoB,Q;QAAhB,w  
BAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,WAAW,UAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IA  
AP,C;;QAEhB,OAAO,W;O;KARX,C;0FAWA,yB;MAAA,gD;MAAA,oD;QAIoB,Q;QAAhB,wBAAgB,SAAhB,g  
B;UAAgB,cAAA,SAAhB,M;UACI,WAAW,UAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OA  
AO,W;O;KARX,C;0FAWA,yB;MAAA,gD;MAAA,oD;QAIoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,S  
AAhB,M;UACI,WAAW,UAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KARX,C;  
0FAWA,yB;MAAA,gD;MAAA,oD;QAIoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,W  
AAW,UAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KARX,C;0FAWA,yB;MAA

A,gD;MAAA,oD;QAIoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,WAAW,UAAU,OAA V,C;UACC,OAAZ,WAAy,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KARX,C;0FAWA,yB;MAAA,oC;MAAA,gD;M AAA,gC;MAAA,oD;QAIoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UACI,WAAW, UAAU,oBAAV,C;UACC,OAAZ,WAAy,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KARX,C;0FAWA,yB;MAAA,gD; MAAA,oD;QAQoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,WAAW,UAAU,OAAV,C; UACC,OAAZ,WAAy,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KAZX,C;0FAeA,yB;MAAA,wE;MAiOA,+D;MAjO A,yC;QASW,kBAAU,oB;QAiOD,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,UAIoiD,WakOvC,CAAY,OAAZ,C;UO p5UP,U;UADP,YPs5Ue,Wot5UH,WPs5UwB,GoT5UxB,C;UACL,IAAI,aAAJ,C;YACH,aPo5UuC,gB;YAA5B,W On5UX,aPm5UgC,GOn5UhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPg5UA,iB;UACA,IAAK,WAAI,OAAJ,C;;Q ApOT,OAsOO,W;O;KA/OX,C;sFAYA,yB;MAAA,wE;MASOA,+D;MATOA,yC;QASW,kBAAU,oB;QAsOD,Q;Q AAhB,iD;UAAgB,cAAhB,e;UACI,UAvOoD,WauO1C,CAAY,OAAZ,C;UOr6UP,U;UADP,YPu6Ue,Wov6UH,W Pu6UwB,GOv6UxB,C;UACL,IAAI,aAAJ,C;YACH,aPq6UuC,gB;YAA5B,WOp6UX,aPo6UgC,GOp6UhC,EAAS, MAAT,C;YACA,e;;YAEA,c;;UPi6UA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAzOT,OA2OO,W;O;KApPX,C;sFAY A,yB;MAAA,wE;MA2OA,+D;MA3OA,yC;QASW,kBAAU,oB;QA2OD,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,U A5OqD,WA4O3C,CAAY,OAAZ,C;UOt7UP,U;UADP,YPw7Ue,Wox7UH,WpW7UwB,GOx7UxB,C;UACL,IAAI, aAAJ,C;YACH,aPs7UuC,gB;YAA5B,WOr7UX,aPq7UgC,GOr7UhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPk7U A,iB;UACA,IAAK,WAAI,OAAJ,C;;QA9OT,OAgPO,W;O;KAZPX,C;sFAYA,yB;MAAA,wE;MAgPA,+D;MAhP A,yC;QASW,kBAAU,oB;QAgPD,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,UAjPmD,WaiPzC,CAAY,OAAZ,C;UO v8UP,U;UADP,YPy8Ue,Woz8UH,Wpy8UwB,GoZ8UxB,C;UACL,IAAI,aAAJ,C;YACH,aPu8UuC,gB;YAA5B, Wot8UX,aPs8UgC,Got8UhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPm8UA,iB;UACA,IAAK,WAAI,OAAJ,C;;Q AnPT,OAqPO,W;O;KA9PX,C;sFAYA,yB;MAAA,wE;MAqPA,+D;MARPA,yC;QASW,kBAAU,oB;QAqPD,Q;QA AhB,iD;UAAgB,cAAhB,e;UACI,UAtPoD,WAsP1C,CAAY,OAAZ,C;Uox9UP,U;UADP,YP09Ue,Wo19UH,Wp0 9UwB,GO19UxB,C;UACL,IAAI,aAAJ,C;YACH,aPw9UuC,gB;YAA5B,Wov9UX,aPu9UgC,GOv9UhC,EAAS,M AAT,C;YACA,e;;YAEA,c;;UPo9UA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAxPT,OA0PO,W;O;KANQX,C;sFAYA, yB;MAAA,wE;MA0PA,+D;MA1PA,yC;QASW,kBAAU,oB;QA0PD,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,UA3 PqD,WA2P3C,CAAY,OAAZ,C;UOz+UP,U;UADP,YP2+Ue,Wo3+UH,Wp2+UwB,GO3+UxB,C;UACL,IAAI,aA AJ,C;YACH,aPy+UuC,gB;YAA5B,Wox+UX,aPw+UgC,GOx+UhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPq+U A,iB;UACA,IAAK,WAAI,OAAJ,C;;QA7PT,OA+PO,W;O;KAXQX,C;sFAYA,yB;MAAA,wE;MA+PA,+D;MA/P A,yC;QASW,kBAAU,oB;QA+PD,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,UAhQsD,WAgQ5C,CAAY,OAAZ,C;U O1/UP,U;UADP,YP4/Ue,Wo5/UH,Wp4/UwB,GO5/UxB,C;UACL,IAAI,aAAJ,C;YACH,aP0/UuC,gB;YAA5B,W Oz/UX,aPy/UgC,GOz/UhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPs/UA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAIQ T,OAoQO,W;O;KA7QX,C;sFAYA,yB;MAAA,wE;MAoQA,+D;MApQA,yC;QASW,kBAAU,oB;QAoQD,Q;QAA hB,iD;UAAgB,cAAhB,e;UACI,UArQuD,WaqQ7C,CAAY,OAAZ,C;UO3gVP,U;UADP,YP6gVe,Wo7gVH,Wp6 gVwB,GO7gVxB,C;UACL,IAAI,aAAJ,C;YACH,aP2gVuC,gB;YAA5B,Wo1gVX,aP0gVgC,GO1gVhC,EAAS,M AAT,C;YACA,e;;YAEA,c;;UPugVA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAvQT,OAYQO,W;O;KAIRX,C;sFAYA, yB;MAAA,wE;MAyQA,oC;MAAA,+D;MAAA,gC;MAzQA,yC;QASW,kBAAU,oB;QAYQD,Q;QAAhB,iD;UAA gB,cAAhB,0B;UACI,UA1QoD,WA0Q1C,CAAY,oBAAZ,C;UO5hVP,U;UADP,YP8hVe,Wo9hVH,Wp8hVwB,G O9hVxB,C;UACL,IAAI,aAAJ,C;YACH,aP4hVuC,gB;YAA5B,Wo3hVX,aP2hVgC,GO3hVhC,EAAS,MAAT,C;Y ACA,e;;YAEA,c;;UPwhVA,iB;UACA,IAAK,WAAI,oBAAJ,C;;QA5QT,OA8QO,W;O;KAvRX,C;sFAYA,yB;MA AAA,wE;MA8QA,+D;MA9QA,yD;QAUW,kBAAU,oB;QA8QD,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,UA/QiD, WA+QvC,CAAY,OAAZ,C;UO9iVP,U;UADP,YPgjVe,WohjVH,WpgjVwB,GOhjVxB,C;UACL,IAAI,aAAJ,C;Y ACH,aP8iVuC,gB;YAA5B,Wo7iVX,aP6iVgC,GO7iVhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UP0iVA,iB;UACA ,IAAK,WajRyD,cAiRrD,CAAE,OAAf,CAAJ,C;;QAJRT,OAmRO,W;O;KA7RX,C;sFAaA,yB;MAAA,wE;MAmR A,+D;MAnRA,yD;QAUW,kBAAU,oB;QAmRD,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,UApRiD,WaORvC,CAA Y,OAAZ,C;UOhkVP,U;UADP,YPkkVe,WolkVH,WpkkVwB,GOlkVxB,C;UACL,IAAI,aAAJ,C;YACH,aPkgVuC ,gB;YAA5B,Wo/jVX,aP+jVgC,GO/jVhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UP4jVA,iB;UACA,IAAK,WatRy D,cAsRrD,CAAE,OAAf,CAAJ,C;;QAtRT,OAwRO,W;O;KAISX,C;uFAaA,yB;MAAA,wE;MAwRA,+D;MAxRA, yD;QAUW,kBAAU,oB;QAwRD,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,UAzRiD,WaYRvC,CAAY,OAAZ,C;UOI

IVP,U;UADP,YPolVe,WOpIVH,WPolVwB,GOpIVxB,C;UACL,IAAI,aAAJ,C;YACH,aPklVuC,gB;YAA5B,Wojl  
VX,aPilVgC,GOjIVhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UP8kVA,iB;UACA,IAAK,WA3RyD,cA2RrD,CAAE,  
OAAf,CAAJ,C;;QA3RT,OA6RO,W;O;KAvSX,C;uFAaA,yB;MAAA,wE;MA6RA,+D;MA7RA,yD;QAUW,kBAA  
U,oB;QA6RD,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,UA9RiD,WA8RvC,CAAY,OAAZ,C;UOpmVP,U;UADP,Y  
PsmVe,WOtmVH,WpSmVwB,GOTmVxB,C;UACL,IAAI,aAAJ,C;YACH,aPomVuC,gB;YAA5B,WOnmVX,aPmm  
VgC,GOnmVhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPgmVA,iB;UACA,IAAK,WahSyD,cAgSrD,CAAE,OAAf,  
CAAJ,C;;QAhST,OAkSO,W;O;KA5SX,C;uFAaA,yB;MAAA,wE;MAkSA,+D;MAISA,yD;QAUW,kBAAU,oB;Q  
AkSD,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,UAnSiD,WAmSvC,CAAY,OAAZ,C;UOtmVP,U;UADP,YPwnVe,W  
OxnVH,WpwnVwB,GOxnVxB,C;UACL,IAAI,aAAJ,C;YACH,aPsnVuC,gB;YAA5B,WOrnVX,aPqnVgC,GOrnV  
hC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPknVA,iB;UACA,IAAK,WArSyD,cAqSrD,CAAE,OAAf,CAAJ,C;;QAr  
ST,OAuSO,W;O;KAjTX,C;uFAaA,yB;MAAA,wE;MAuSA,+D;MAvSA,yD;QAUW,kBAAU,oB;QAUdSD,Q;QA  
AhB,iD;UAAgB,cAAhB,e;UACI,UAxSiD,WAwSvC,CAAY,OAAZ,C;UOxoVP,U;UADP,YP0oVe,W01oVH,WP0o  
VwB,GO1oVxB,C;UACL,IAAI,aAAJ,C;YACH,aPwoVuC,gB;YAA5B,W0voVX,aPuoVgC,GOvoVhC,EAAS,MA  
AT,C;YACA,e;;YAEA,c;;UPooVA,iB;UACA,IAAK,WA1SyD,cA0SrD,CAAE,OAAf,CAAJ,C;;QA1ST,OA4SO,W  
;O;KAiTX,C;uFAaA,yB;MAAA,wE;MA4SA,+D;MA5SA,yD;QAUW,kBAAU,oB;QA4SD,Q;QAAhB,iD;UAAgB,  
cAAhB,e;UACI,UA7SiD,WA6SvC,CAAY,OAAZ,C;UO1pVP,U;UADP,YP4pVe,W05pVH,WP4pVwB,GO5pVx  
B,C;UACL,IAAI,aAAJ,C;YACH,aP0pVuC,gB;YAA5B,W0zpVX,aPypVgC,GOzpVhC,EAAS,MAAT,C;YACA,e;  
;YAEA,c;;UPspVA,iB;UACA,IAAK,WA/SyD,cA+SrD,CAAE,OAAf,CAAJ,C;;QA/ST,OAiTO,W;O;KA3TX,C;uF  
AaA,yB;MAAA,wE;MAiTA,+D;MAjTA,yD;QAUW,kBAAU,oB;QaiTD,Q;QAAhB,iD;UAAgB,cAAhB,e;UACI,  
UAIiTD,WakTvC,CAAY,OAAZ,C;UO5qVP,U;UADP,YP8qVe,W09qVH,WP8qVwB,GO9qVxB,C;UACL,IAAI,  
aAAJ,C;YACH,aP4qVuC,gB;YAA5B,W03qVX,aP2qVgC,GO3qVhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPwq  
VA,iB;UACA,IAAK,WApTyD,cAoTrD,CAAE,OAAf,CAAJ,C;;QApTT,OAStO,W;O;KAhUX,C;uFAaA,yB;MAA  
A,wE;MAStA,oC;MAAA,+D;MAAA,gC;MAiTA,yD;QAUW,kBAAU,oB;QAsTD,Q;QAAhB,iD;UAAgB,cAAhB,  
0B;UACI,UAvTiD,WAvTvC,CAAY,oBAAZ,C;UO9rVP,U;UADP,YPgsVe,W0hsVH,WPgsVwB,GOhsVxB,C;U  
ACL,IAAI,aAAJ,C;YACH,aP8rVuC,gB;YAA5B,W07rVX,aP6rVgC,GO7rVhC,EAAS,MAAT,C;YACA,e;;YAEA  
,c;;UP0rVA,iB;UACA,IAAK,WazTyD,cAyTrD,CAAE,oBAaf,CAAJ,C;;QazTT,OA2TO,W;O;KArUX,C;uFAaA,  
yB;MAAA,+D;MAAA,sD;QASoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,UAAU,YA  
AY,OAAZ,C;UOp5UP,U;UADP,YPs5Ue,W0t5UH,WP5UwB,GOt5UxB,C;UACL,IAAI,aAAJ,C;YACH,aPo5Uu  
C,gB;YAA5B,WOn5UX,aPm5UgC,GOn5UhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPg5UA,iB;UACA,IAAK,W  
AAI,OAAJ,C;;QAET,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,+D;MAAA,sD;QASoB,Q;QAAhB,wBAAgB,SA  
AhB,gB;UAAgB,cAAA,SAAhB,M;UACI,UAAU,YAAY,OAAZ,C;UOr6UP,U;UADP,YPu6Ue,W0v6UH,WPu6U  
wB,GOv6UxB,C;UACL,IAAI,aAAJ,C;YACH,aPq6UuC,gB;YAA5B,WOp6UX,aPo6UgC,GOp6UhC,EAAS,MAA  
T,C;YACA,e;;YAEA,c;;UPI6UA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAET,OAAO,W;O;KAdX,C;0FAiBA,yB;MA  
AA,+D;MAAA,sD;QASoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,UAAU,YAAY,OA  
AZ,C;UOt7UP,U;UADP,YPw7Ue,W0x7UH,WPw7UwB,GOx7UxB,C;UACL,IAAI,aAAJ,C;YACH,aPs7UuC,gB;  
YAA5B,WOr7UX,aPq7UgC,GO7UhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPk7UA,iB;UACA,IAAK,WAAI,O  
AAJ,C;;QAET,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,+D;MAAA,sD;QASoB,Q;QAAhB,wBAAgB,SAAhB,g  
B;UAAgB,cAAA,SAAhB,M;UACI,UAAU,YAAY,OAAZ,C;UOv8UP,U;UADP,YPy8Ue,W0z8UH,WPY8UwB,G  
Oz8UxB,C;UACL,IAAI,aAAJ,C;YACH,aPu8UuC,gB;YAA5B,W0t8UX,aPs8UgC,GOt8UhC,EAAS,MAAT,C;YA  
CA,e;;YAEA,c;;UPm8UA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAET,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,+  
D;MAAA,sD;QASoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,UAAU,YAAY,OAAZ,C;  
UOx9UP,U;UADP,YP09Ue,W019UH,WP09UwB,GO19UxB,C;UACL,IAAI,aAAJ,C;YACH,aPw9UuC,gB;YAA5  
B,W0v9UX,aPu9UgC,GOv9UhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPo9UA,iB;UACA,IAAK,WAAI,OAAJ,C  
;;QAET,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,+D;MAAA,sD;QASoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAA  
gB,cAAA,SAAhB,M;UACI,UAAU,YAAY,OAAZ,C;UOz+UP,U;UADP,YP2+Ue,W03+UH,WP2+UwB,GO3+Ux  
B,C;UACL,IAAI,aAAJ,C;YACH,aPy+UuC,gB;YAA5B,W0x+UX,aPw+UgC,GOx+UhC,EAAS,MAAT,C;YACA,  
e;;YAEA,c;;UPq+UA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAET,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,+D;M  
AAA,sD;QASoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,UAAU,YAAY,OAAZ,C;UO

1/UP,U;UADP,YP4/Ue,WO5/UH,WP4/UwB,GO5/UxB,C;UACL,IAAI,aAAJ,C;YACH,aP0/UuC,gB;YAA5B,WOz  
/UX,aPy/UgC,GOz/UhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPs/UA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAET,O  
AAO,W;O;KAdX,C;0FAiBA,yB;MAAA,+D;MAAA,sD;QASoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,  
SAAhB,M;UACI,UAAU,YAAY,OAAZ,C;UO3gVP,U;UADP,YP6gVe,WO7gVH,WP6gVwB,GO7gVxB,C;UACL  
,IAAI,aAAJ,C;YACH,aP2gVuC,gB;YAA5B,WO1gVX,aP0gVgC,GO1gVhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;  
UPugVA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAET,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,oC;MAAA,+D;MA  
AA,gC;MAAA,sD;QASoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UACI,UAAU,Y  
AAY,oBAAZ,C;UO5hVP,U;UADP,YP8hVe,WO9hVH,WP8hVwB,GO9hVxB,C;UACL,IAAI,aAAJ,C;YACH,aP4  
hVuC,gB;YAA5B,WO3hVX,aP2hVgC,GO3hVhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPwhVA,iB;UACA,IAA  
K,WAAI,oBAAJ,C;;QAET,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,+D;MAAA,sE;QAUoB,Q;QAAhB,wBAAg  
B,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,UAAU,YAAY,OAAZ,C;UO9iVP,U;UADP,YPgjVe,WOhjVH,WP  
gjVwB,GOhjVxB,C;UACL,IAAI,aAAJ,C;YACH,aP8iVuC,gB;YAA5B,WO7iVX,aP6iVgC,GO7iVhC,EAAS,MA  
AT,C;YACA,e;;YAEA,c;;UP0iVA,iB;UACA,IAAK,WAAI,eAAe,OAAf,CAAJ,C;;QAET,OAAO,W;O;KafX,C;0F  
AkBA,yB;MAAA,+D;MAAA,sE;QAUoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,UA  
AU,YAAY,OAAZ,C;UOhkVP,U;UADP,YPkkVe,WOlkVH,WPkkVwB,GOlkVxB,C;UACL,IAAI,aAAJ,C;YACH,  
aPkgVuC,gB;YAA5B,WOjVX,aP+jVgC,GOjVhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UP4jVA,iB;UACA,IAA  
K,WAAI,eAAe,OAAf,CAAJ,C;;QAET,OAAO,W;O;KafX,C;2FAkBA,yB;MAAA,+D;MAAA,sE;QAUoB,Q;QAA  
hB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,UAAU,YAAY,OAAZ,C;UOllVP,U;UADP,YPolVe,W  
OplVH,WPolVwB,GOplVxB,C;UACL,IAAI,aAAJ,C;YACH,aPklVuC,gB;YAA5B,WOjlVX,aPilVgC,GOjlVhC,E  
AAS,MAAT,C;YACA,e;;YAEA,c;;UP8kVA,iB;UACA,IAAK,WAAI,eAAe,OAAf,CAAJ,C;;QAET,OAAO,W;O;K  
AfX,C;2FAkBA,yB;MAAA,+D;MAAA,sE;QAUoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;  
UACI,UAAU,YAAY,OAAZ,C;UOpmVP,U;UADP,YPsmVe,WotmVH,WPsmVwB,GOtmVxB,C;UACL,IAAI,aA  
AJ,C;YACH,aPomVuC,gB;YAA5B,WOnmVX,aPmmVgC,GOnmVhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPgm  
VA,iB;UACA,IAAK,WAAI,eAAe,OAAf,CAAJ,C;;QAET,OAAO,W;O;KafX,C;2FAkBA,yB;MAAA,+D;MAAA,  
sE;QAUoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,UAAU,YAAY,OAAZ,C;UOtnVP,  
U;UADP,YPwnVe,WOxnVH,WPwnVwB,GOxnVxB,C;UACL,IAAI,aAAJ,C;YACH,aPsnVuC,gB;YAA5B,WOrn  
VX,aPqnVgC,GOrnVhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPknVA,iB;UACA,IAAK,WAAI,eAAe,OAAf,CAA  
J,C;;QAET,OAAO,W;O;KafX,C;2FAkBA,yB;MAAA,+D;MAAA,sE;QAUoB,Q;QAAhB,wBAAgB,SAAhB,gB;U  
AAgB,cAAA,SAAhB,M;UACI,UAAU,YAAY,OAAZ,C;UOxoVP,U;UADP,YP0oVe,WO1oVH,WP0oVwB,GO1o  
VxB,C;UACL,IAAI,aAAJ,C;YACH,aPwoVuC,gB;YAA5B,WOvoVX,aPuoVgC,GOvoVhC,EAAS,MAAT,C;YAC  
A,e;;YAEA,c;;UPooVA,iB;UACA,IAAK,WAAI,eAAe,OAAf,CAAJ,C;;QAET,OAAO,W;O;KafX,C;2FAkBA,yB;  
MAAA,+D;MAAA,sE;QAUoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,UAAU,YAAY,  
OAAZ,C;UO1pVP,U;UADP,YP4pVe,WO5pVH,WP4pVwB,GO5pVxB,C;UACL,IAAI,aAAJ,C;YACH,aP0pVuC,  
gB;YAA5B,WOzpVX,aPypVgC,GOzpVhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UPspVA,iB;UACA,IAAK,WAAI  
,eAAe,OAAf,CAAJ,C;;QAET,OAAO,W;O;KafX,C;2FAkBA,yB;MAAA,+D;MAAA,sE;QAUoB,Q;QAAhB,wBA  
AgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,UAAU,YAAY,OAAZ,C;UO5qVP,U;UADP,YP8qVe,WO9qVH  
,WP8qVwB,GO9qVxB,C;UACL,IAAI,aAAJ,C;YACH,aP4qVuC,gB;YAA5B,WO3qVX,aP2qVgC,GO3qVhC,EA  
AS,MAAT,C;YACA,e;;YAEA,c;;UPwqVA,iB;UACA,IAAK,WAAI,eAAe,OAAf,CAAJ,C;;QAET,OAAO,W;O;K  
AfX,C;2FAkBA,yB;MAAA,oC;MAAA,+D;MAAA,gC;MAAA,sE;QAUoB,Q;QAAhB,wBAAgB,SAAhB,gB;UAA  
gB,cAAhB,UAAgB,SAAhB,O;UACI,UAAU,YAAY,oBAAZ,C;UO9rVP,U;UADP,YPgsVe,WOhsVH,WPgsVwB,  
GOhsVxB,C;UACL,IAAI,aAAJ,C;YACH,aP8rVuC,gB;YAA5B,WO7rVX,aP6rVgC,GO7rVhC,EAAS,MAAT,C;Y  
ACA,e;;YAEA,c;;UP0rVA,iB;UACA,IAAK,WAAI,eAAe,oBAAf,CAAJ,C;;QAET,OAAO,W;O;KafX,C;0FAkBA  
,yB;MAAA,kC;MAAA,4C;MAAA,wE;QAQW,sC;QAAA,8C;O;MARX,oDASQ,Y;QAA6C,OAAgB,qBAAhB,oB  
AAgB,C;O;MATrE,iDAUQ,mB;QAAoC,gCAAY,OAAZ,C;O;MAV5C,gF;MAAA,yC;QAQI,2D;O;KARJ,C;4EA  
cA,yB;MAAA,gE;MAAA,uC;QAOW,kBAAM,eAAa,gBAAb,C;QA+UA,Q;QAAb,iD;UAAa,WAAb,e;UACI,WAA  
Y,WAhViB,SAgVb,CAAU,IAAV,CAAJ,C;;QAhVhB,OaiVO,W;O;KaxVX,C;8EAUA,yB;MAAA,gE;MAAA,uC  
;QAOW,kBAAM,eAAa,gBAAb,C;QA+UA,Q;QAAb,iD;UAAa,WAAb,e;UACI,WAAy,WAhViB,SAgVb,CAAU,I  
AAV,CAAJ,C;;QAhVhB,OaiVO,W;O;KaxVX,C;8EAUA,yB;MAAA,gE;MAAA,uC;QAOW,kBAAM,eAAa,gBA

Ab,C;QA+UA,Q;QAAb,iD;UAAa,WAAb,e;UACI,WAAy,WAhViB,SAGVb,CAAU,IAAV,CAAJ,C;;QAhVhB,OA  
iVO,W;O;KAxVX,C;8EAUA,yB;MAAA,gE;MAAA,uC;QAOW,kBAAM,eAAa,gBAAb,C;QA+UA,Q;QAAb,iD;  
UAAa,WAAb,e;UACI,WAAy,WAhViB,SAGVb,CAAU,IAAV,CAAJ,C;;QAhVhB,OAiVO,W;O;KAxVX,C;8EAU  
A,yB;MAAA,gE;MAAA,uC;QAOW,kBAAM,eAAa,gBAAb,C;QA+UA,Q;QAAb,iD;UAAa,WAAb,e;UACI,WAA  
Y,WAhViB,SAGVb,CAAU,IAAV,CAAJ,C;;QAhVhB,OAiVO,W;O;KAxVX,C;8EAUA,yB;MAAA,gE;MAAA,uC  
;QAOW,kBAAM,eAAa,gBAAb,C;QA+UA,Q;QAAb,iD;UAAa,WAAb,e;UACI,WAAy,WAhViB,SAGVb,CAAU,I  
AAV,CAAJ,C;;QAhVhB,OAiVO,W;O;KAxVX,C;8EAUA,yB;MAAA,gE;MAAA,uC;QAOW,kBAAM,eAAa,gBA  
Ab,C;QA+UA,Q;QAAb,iD;UAAa,WAAb,e;UACI,WAAy,WAhViB,SAGVb,CAAU,IAAV,CAAJ,C;;QAhVhB,OA  
iVO,W;O;KAxVX,C;8EAUA,yB;MAAA,gE;MAAA,uC;QAOW,kBAAM,eAAa,gBAAb,C;QA+UA,Q;QAAb,iD;  
UAAa,WAAb,e;UACI,WAAy,WAhViB,SAGVb,CAAU,IAAV,CAAJ,C;;QAhVhB,OAiVO,W;O;KAxVX,C;8EAU  
A,yB;MAAA,gE;MAIvA,oC;MAAA,gC;MAjvA,uC;QAOW,kBAAM,eAAa,gBAAb,C;QA+UA,Q;QAAb,iD;UA  
Aa,WAAb,0B;UACI,WAAy,WAhViB,SAGVb,CAAU,iBAAV,CAAJ,C;;QAhVhB,OAiVO,W;O;KAxVX,C;0FAU  
A,yB;MAAA,gE;MAAA,uC;QAOW,kBAaA,eAAa,gBAAb,C;QAghP,gB;QADb,YAAY,C;QACZ,iD;UAAa,WA  
Ab,e;UACI,WAAy,WAjHwB,SAiHpB,EAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;QAJHhB,OAkH  
O,W;O;KAzHX,C;4FAUA,yB;MAAA,gE;MAAA,uC;QAOW,kBAaA,eAAa,gBAAb,C;QAmHP,gB;QADb,YAAY  
,C;QACZ,iD;UAAa,WAAb,e;UACI,WAAy,WApHwB,SAoHpB,EAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,  
CAAJ,C;;QApHhB,OAqHO,W;O;KA5HX,C;4FAUA,yB;MAAA,gE;MAAA,uC;QAOW,kBAaA,eAAa,gBAAb,C;  
QAsHP,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,e;UACI,WAAy,WAvHwB,SAuHpB,EAAU,cAAV,EAAU,s  
BAAV,WAAmB,IAAnB,CAAJ,C;;QAvHhB,OAwhO,W;O;KA/HX,C;4FAUA,yB;MAAA,gE;MAAA,uC;QAOW,  
kBAaA,eAAa,gBAAb,C;QAyHP,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,e;UACI,WAAy,WA1HwB,SA0Hp  
B,EAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;QA1HhB,OA2HO,W;O;KA1IX,C;4FAUA,yB;MAAA,  
gE;MAAA,uC;QAOW,kBAaA,eAAa,gBAAb,C;QA4HP,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,e;UACI,W  
AAy,WA7HwB,SA6HpB,EAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;QA7HhB,OA8HO,W;O;KArl  
X,C;2FAUA,yB;MAAA,gE;MAAA,uC;QAOW,kBAaA,eAAa,gBAAb,C;QA+HP,gB;QADb,YAAY,C;QACZ,iD;U  
AAa,WAAb,e;UACI,WAAy,WAhIwB,SAGIpB,EAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;QAhIhB  
,OAiIO,W;O;KAxIX,C;4FAUA,yB;MAAA,gE;MAAA,uC;QAOW,kBAaA,eAAa,gBAAb,C;QAkIP,gB;QADb,YA  
AY,C;QACZ,iD;UAAa,WAAb,e;UACI,WAAy,WAnIwB,SAmIpB,EAAU,cAAV,EAAU,sBAAV,WAAmB,IAAn  
B,CAAJ,C;;QAnIhB,OAoIO,W;O;KA3IX,C;4FAUA,yB;MAAA,gE;MAAA,uC;QAOW,kBAaA,eAAa,gBAAb,C;  
QAqIP,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,e;UACI,WAAy,WAtIwB,SASIpB,EAAU,cAAV,EAAU,sBA  
AV,WAAmB,IAAnB,CAAJ,C;;QAtIhB,OAuIO,W;O;KA9IX,C;4FAUA,yB;MAAA,gE;MAuIA,oC;MAAA,gC;M  
AvIA,uC;QAOW,kBAaA,eAAa,gBAAb,C;QAwIP,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,0B;UACI,WAAy  
,WAZIwB,SAyIpB,EAAU,cAAV,EAAU,sBAAV,WAAmB,iBAAnB,CAAJ,C;;QAzIhB,OA0IO,W;O;KAjJX,C;wG  
AUA,yB;MAAA,+D;MAAA,uC;QAOW,kBAaOb,gB;QA8iEd,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,e;UA  
piEmC,U;UAAA,cAVQ,SAUR,EAoiET,cApiES,EAoiET,sBApiES,WAOiEA,IApiEA,W;YAA6C,6B;;QAVhF,OA  
WO,W;O;KAIBX,C;4GAUA,yB;MAAA,oD;QA2iEiB,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,e;UApiEmC,  
U;UAAA,yBAoiET,cApiES,EAoiET,sBApiES,WAOiEA,IApiEA,W;YAA6C,6B;;QACHf,OAAO,W;O;KARX,C;8  
FAWA,6C;MAQiB,UACiB,M;MAF9B,YAAY,C;MACZ,wBAaA,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,WAA  
Y,WAAI,WAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;MACHB,OAAO,W;K;gGAGX,6C;MAQiB,U  
ACiB,M;MAF9B,YAAY,C;MACZ,wBAaA,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,WAAy,WAAI,WAAU,cAAV,EAAU,sBAAV,WA  
AmB,IAAnB,CAAJ,C;;MACHB,OAAO,W;K;gGAGX,6C;MAQiB,UACiB,M;MAF9B,YAAY,C;MACZ,wBAaA,SA  
Ab,gB;QAAa,WAAA,SAAb,M;QACI,WAAy,WAAI,WAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;  
MACHB,OAAO,W;K;gGAGX,6C;MAQiB,UACiB,M;MAF9B,YAAY,C;MACZ,wBAaA,SAAb,gB;QAAa,WAAA  
,SAAb,M;QACI,WAAy,WAAI,WAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;MACHB,OAAO,W;K;g  
GAGX,6C;MAQiB,UACiB,M;MAF9B,YAAY,C;MACZ,wBAaA,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,WAA  
Y,WAAI,WAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;MACHB,OAAO,W;K;gGAGX,6C;MAQiB,U  
ACiB,M;MAF9B,YAAY,C;MACZ,wBAaA,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,WAAy,WAAI,WAAU,cAA

V,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;MACHB,OAAO,W;K;+FAGX,6C;MAQiB,UACiB,M;MAF9B,YAA  
Y,C;MACZ,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,WAAy,WAAI,WAAU,cAAV,EAAU,sBAAV,WA  
AmB,IAAnB,CAAJ,C;;MACHB,OAAO,W;K;gGAGX,yB;MAAA,oC;MAAA,gC;MAAA,oD;QAQiB,UACiB,M;Q  
AF9B,YAAy,C;QACZ,wBAAa,SAAb,gB;UAAa,WAAb,UAAa,SAAb,O;UACI,WAAy,WAAI,WAAU,cAAV,EA  
AU,sBAAV,WAAmB,iBAAnB,CAAJ,C;;QACHB,OAAO,W;O;KAVX,C;0FAaA,yB;MAAA,+D;MAAA,uC;QAO  
W,kBAAa,gB;Qak2DJ,Q;QAaHb,iD;UAAgB,cAAhB,e;UA11DqB,U;UAAA,cARe,SAQf,CA01DQ,OA11DR,W;  
YAAc,6B;;QAR3D,OASO,W;O;KAhBX,C;8FAUA,yB;MAAA,oD;QA+1DoB,Q;QAaHb,iD;UAAgB,cAAhB,e;  
UA11DqB,U;UAAA,wBA01DQ,OA11DR,W;YAAc,6B;;QAC3D,OAAO,W;O;KANX,C;gFASA,6C;MAKiB,Q;  
MAAb,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,WAAy,WAAI,UAAU,IAAV,CAAJ,C;;MACHB,OAAO,  
W;K;kFAGX,6C;MAKiB,Q;MAAb,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,WAAy,WAAI,UAAU,IAA  
V,CAAJ,C;;MACHB,OAAO,W;K;kFAGX,6C;MAKiB,Q;MAAb,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QAC  
I,WAAy,WAAI,UAAU,IAAV,CAAJ,C;;MACHB,OAAO,W;K;kFAGX,6C;MAKiB,Q;MAAb,wBAAa,SAAb,gB;  
QAAa,WAAA,SAAb,M;QACI,WAAy,WAAI,UAAU,IAAV,CAAJ,C;;MACHB,OAAO,W;K;kFAGX,6C;MAKiB,  
Q;MAAb,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,WAAy,WAAI,UAAU,IAAV,CAAJ,C;;MACHB,OAA  
O,W;K;kFAGX,6C;MAKiB,Q;MAAb,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,WAAy,WAAI,UAAU,IA  
AV,CAAJ,C;;MACHB,OAAO,W;K;kFAGX,6C;MAKiB,Q;MAAb,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QA  
CI,WAAy,WAAI,UAAU,IAAV,CAAJ,C;;MACHB,OAAO,W;K;kFAGX,6C;MAKiB,Q;MAAb,wBAAa,SAAb,gB;  
QAAa,WAAA,SAAb,M;QACI,WAAy,WAAI,UAAU,IAAV,CAAJ,C;;MACHB,OAAO,W;K;kFAGX,yB;MAAA,o  
C;MAAA,gC;MAAA,oD;QAKiB,Q;QAAb,wBAAa,SAAb,gB;UAAa,WAAb,UAAa,SAAb,O;UACI,WAAy,WAA  
I,UAAU,iBAAV,CAAJ,C;;QACHB,OAAO,W;O;KAPX,C;IAe4B,0C;MAAA,mB;QAAE,2C;O;K;IAL9B,8B;MAK  
I,OAAO,qBAaiB,2BAAjB,C;K;IAQiB,4C;MAAA,mB;QAAE,+C;O;K;IAL9B,gC;MAKI,OAAO,qBAaiB,6BAAj  
B,C;K;IAQiB,4C;MAAA,mB;QAAE,gD;O;K;IAL9B,gC;MAKI,OAAO,qBAaiB,6BAAjB,C;K;IAQiB,4C;MAAA,  
mB;QAAE,8C;O;K;IAL9B,gC;MAKI,OAAO,qBAaiB,6BAAjB,C;K;IAQiB,4C;MAAA,mB;QAAE,+C;O;K;IAL9  
B,gC;MAKI,OAAO,qBAaiB,6BAAjB,C;K;IAQiB,4C;MAAA,mB;QAAE,gD;O;K;IAL9B,gC;MAKI,OAAO,qBA  
AiB,6BAAjB,C;K;IAQiB,4C;MAAA,mB;QAAE,iD;O;K;IAL9B,gC;MAKI,OAAO,qBAaiB,6BAAjB,C;K;IAQiB,  
4C;MAAA,mB;QAAE,kD;O;K;IAL9B,gC;MAKI,OAAO,qBAaiB,6BAAjB,C;K;IAQiB,4C;MAAA,mB;QAAE,+  
C;O;K;IAL9B,gC;MAKI,OAAO,qBAaiB,6BAAjB,C;K;IAGX,6B;MASI,OAA2B,SAAf,aAAL,SAAK,CAAe,C;K;  
IAG/B,+B;MAQI,OAA2B,SAAf,eAAL,SAAK,CAAe,C;K;IAG/B,+B;MAQI,OAA2B,SAAf,eAAL,SAAK,CAAe,C  
;K;IAG/B,+B;MAQI,OAA2B,SAAf,eAAL,SAAK,CAAe,C;K;IAG/B,+B;MAQI,OAA2B,SAAf,eAAL,SAAK,CAA  
e,C;K;IAG/B,+B;MAQI,OAA2B,SAAf,eAAL,SAAK,CAAe,C;K;IAG/B,+B;MAQI,OAA2B,SAAf,eAAL,SAAK,CAA  
e,C;K;IAG/B,+B;MAQI,OAA2B,SAAf,eAAL,SAAK,CAAe,C;K;IAG/B,+B;MAQI,OAA2B,SAAf,eAAL,SAAK,CAA  
K,CAAe,C;K;0FAG/B,yB;MAAA,2D;MAAA,+D;MAAA,sC;QAYc,Q;QAFV,UAAU,c;QACV,WAAW,gB;QAC  
X,wBAAU,SAAV,gB;UAAU,QAAA,SAAV,M;UACI,UAAU,SAAS,CAAT,C;UACV,IAAI,GAAL,WAAI,GAAL,CAA  
AR,C;YACI,IAAK,WAAI,CAAJ,C;;QAEb,OAAO,I;O;KAjBX,C;4FAoBA,yB;MAAA,2D;MAAA,+D;MAAA,s  
C;QAWc,Q;QAFV,UAAU,c;QACV,WAAW,gB;QACX,wBAAU,SAAV,gB;UAAU,QAAA,SAAV,M;UACI,UAA  
U,SAAS,CAAT,C;UACV,IAAI,GAAL,WAAI,GAAL,CAAJ,CAAR,C;YACI,IAAK,WAAI,CAAJ,C;;QAEb,OAAO,I;O;K  
AhBX,C;4FAmBA,yB;MAAA,2D;MAAA,+D;MAAA,sC;QAWc,Q;QAFV,UAAU,c;QACV,WAAW,gB;QACX,w  
BAAU,SAAV,gB;UAAU,QAAA,SAAV,M;UACI,UAAU,SAAS,CAAT,C;UACV,IAAI,GAAL,WAAI,GAAL,CAA  
R,C;YACI,IAAK,WAAI,CAAJ,C;;QAEb,OAAO,I;O;KAhBX,C;4FAmBA,yB;MAAA,2D;MAAA,+D;MAAA,sC;  
QAWc,Q;QAFV,UAAU,c;QACV,WAAW,gB;QACX,wBAAU,SAAV,gB;UAAU,QAAA,SAAV,M;UACI,UAAU,  
SAAS,CAAT,C;UACV,IAAI,GAAL,WAAI,GAAL,CAAJ,CAAR,C;YACI,IAAK,WAAI,CAAJ,C;;QAEb,OAAO,I;O;KAh  
BX,C;4FAmBA,yB;MAAA,2D;MAAA,+D;MAAA,sC;QAWc,Q;QAFV,UAAU,c;QACV,WAAW,gB;QACX,wBA  
AU,SAAV,gB;UAAU,QAAA,SAAV,M;UACI,UAAU,SAAS,CAAT,C;UACV,IAAI,GAAL,WAAI,GAAL,CAAJ,CAAR,C  
;YACI,IAAK,WAAI,CAAJ,C;;QAEb,OAAO,I;O;KAhBX,C;4FAmBA,yB;MAAA,2D;MAAA,+D;MAAA,sC;QA  
Wc,Q;QAFV,UAAU,c;QACV,WAAW,gB;QACX,wBAAU,SAAV,gB;UAAU,QAAA,SAAV,M;UACI,UAAU,SA  
AS,CAAT,C;UACV,IAAI,GAAL,WAAI,GAAL,CAAJ,CAAR,C;YACI,IAAK,WAAI,CAAJ,C;;QAEb,OAAO,I;O;KAhBX  
,C;4FAmBA,yB;MAAA,2D;MAAA,+D;MAAA,sC;QAWc,Q;QAFV,UAAU,c;QACV,WAAW,gB;QACX,wBAA  
U,SAAV,gB;UAAU,QAAA,SAAV,M;UACI,UAAU,SAAS,CAAT,C;UACV,IAAI,GAAL,WAAI,GAAL,CAAJ,CAAR,C;

YACI,IAAK,WAAI,CAAJ,C;;QAEb,OAAO,I;O;KAhBX,C;4FAMBA,yB;MAAA,2D;MAAA,+D;MAAA,sC;QA  
Wc,Q;QAFV,UAAU,c;QACV,WAAW,gB;QACX,wBAAU,SAAV,gB;UAAU,QAAA,SAAV,M;UACI,UAAU,SA  
AS,CAAT,C;UACV,IAAI,GAAL,WAAI,GAAL,CAAR,C;YACI,IAAK,WAAI,CAAJ,C;;QAEb,OAAO,I;O;KAhBX  
,C;4FAMBA,yB;MAAA,2D;MAAA,+D;MAAA,oC;MAAA,gC;MAAA,sC;QAWc,Q;QAFV,UAAU,c;QACV,WA  
AW,gB;QACX,wBAAU,SAAV,gB;UAAU,QAAV,UAAU,SAAV,O;UACI,UAAU,SAAS,cAAT,C;UACV,IAAI,GA  
AAI,WAAI,GAAL,CAAR,C;YACI,IAAK,WAAI,cAAJ,C;;QAEb,OAAO,I;O;KAhBX,C;IAmBA,qC;MAQI,UAAe,  
aAAL,SAAK,C;MACX,YAAJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;K;IAGX,uC;MAQI,UAAe,eAAL,SAAK,  
C;MACX,YAAJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;K;IAGX,uC;MAQI,UAAe,eAAL,SAAK,C;MACX,YA  
AJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;K;IAGX,uC;MAQI,UAAe,eAAL,SAAK,C;MACX,YAAJ,GAAL,EAA  
U,KAAV,C;MACJ,OAAO,G;K;IAGX,uC;MAQI,UAAe,eAAL,SAAK,C;MACX,YAAJ,GAAL,EAAU,KAAV,C;M  
ACJ,OAAO,G;K;IAGX,uC;MAQI,UAAe,eAAL,SAAK,C;MACX,YAAJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;  
K;IAGX,uC;MAQI,UAAe,eAAL,SAAK,C;MACX,YAAJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;K;IAGX,uC;M  
AQI,UAAe,eAAL,SAAK,C;MACX,YAAJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;K;IAGX,uC;MAQI,UAAe,eA  
AL,SAAK,C;MACX,YAAJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;K;IAGX,oC;MAMI,UAAe,aAAL,SAAK,C;  
MACX,YAAJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;K;IAGX,sC;MAMI,UAAe,eAAL,SAAK,C;MACX,YAAJ,  
GAAL,EAAU,KAAV,C;MACJ,OAAO,G;K;IAGX,sC;MAMI,UAAe,eAAL,SAAK,C;MACX,YAAJ,GAAL,EAAU,  
KAAV,C;MACJ,OAAO,G;K;IAGX,sC;MAMI,UAAe,eAAL,SAAK,C;MACX,YAAJ,GAAL,EAAU,KAAV,C;MA  
CJ,OAAO,G;K;IAGX,sC;MAMI,UAAe,eAAL,SAAK,C;MACX,YAAJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;  
K;IAGX,sC;MAMI,UAAe,eAAL,SAAK,C;MACX,YAAJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;K;IAGX,sC;M  
AMI,UAAe,eAAL,SAAK,C;MACX,YAAJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;K;IAGX,sC;MAMI,UAAe,e  
AAL,SAAK,C;MACX,YAAJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;K;IAGX,sC;MAMI,UAAe,eAAL,SAAK,C  
;MACX,YAAJ,GAAL,EAAU,KAAV,C;MACJ,OAAO,G;K;IAGX,iC;MAMI,OAAO,wBAaA,qBAaIB,YAAy,gB  
AAZ,CAAJB,CAAb,C;K;IAGX,mC;MAMI,OAAO,0BAaA,qBAaOB,YAAy,gBAAZ,CAApB,CAAb,C;K;IAGX,  
mC;MAMI,OAAO,0BAaA,qBAaQB,YAAy,gBAAZ,CAArB,CAAb,C;K;IAGX,mC;MAMI,OAAO,0BAaA,qBAa  
mB,YAAy,gBAAZ,CAAnB,CAAb,C;K;IAGX,mC;MAMI,OAAO,0BAaA,qBAaOB,YAAy,gBAAZ,CAApB,CA  
Ab,C;K;IAGX,mC;MAMI,OAAO,0BAaA,qBAaQB,YAAy,gBAAZ,CAArB,CAAb,C;K;IAGX,mC;MAMI,OAAO  
,0BAaA,qBAaSB,YAAy,gBAAZ,CAAtB,CAAb,C;K;IAGX,mC;MAMI,OAAO,0BAaA,qBAaUB,YAAy,gBAAZ  
,CAAvB,CAAb,C;K;IAGX,mC;MAMI,OAAO,0BAaA,qBAaOB,YAAiB,eAAL,gBAAK,EAAa,GAAb,CAAJB,CA  
ApB,CAAb,C;K;IAGX,iC;MAUI,UAAe,aAAL,SAAK,C;MACX,OAAJ,GAAL,EAAO,KAAP,C;MACJ,OAAO,G;  
K;IAGX,mC;MAUI,UAAe,eAAL,SAAK,C;MACX,OAAJ,GAAL,EAAO,KAAP,C;MACJ,OAAO,G;K;IAGX,mC;  
MAUI,UAAe,eAAL,SAAK,C;MACX,OAAJ,GAAL,EAAO,KAAP,C;MACJ,OAAO,G;K;IAGX,mC;MAUI,UAAe,  
eAAL,SAAK,C;MACX,OAAJ,GAAL,EAAO,KAAP,C;MACJ,OAAO,G;K;IAGX,mC;MAUI,UAAe,eAAL,SAAK,  
C;MACX,OAAJ,GAAL,EAAO,KAAP,C;MACJ,OAAO,G;K;IAGX,mC;MAUI,UAAe,eAAL,SAAK,C;MACX,OA  
AJ,GAAL,EAAO,KAAP,C;MACJ,OAAO,G;K;IAGX,mC;MAUI,UAAe,eAAL,SAAK,C;MACX,OAAJ,GAAL,EA  
AO,KAAP,C;MACJ,OAAO,G;K;IAGX,mC;MAUI,UAAe,eAAL,SAAK,C;MACX,OAAJ,GAAL,EAAO,KAAP,C;  
MACJ,OAAO,G;K;IAGX,mC;MAUI,UAAe,eAAL,SAAK,C;MACX,OAAJ,GAAL,EAAO,KAAP,C;MACJ,OAAO,  
G;K;4EAGX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QAAsB,IAAI,CAAC,UA  
AU,OAAV,CAAL,C;UAAyB,OAAO,K;;MACtD,OAAO,I;K;8EAGX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,  
gB;QAAGB,cAAA,SAAhB,M;QAAsB,IAAI,CAAC,UAAU,OAAV,CAAL,C;UAAyB,OAAO,K;;MACtD,OAAO,I  
;K;8EAGX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QAAsB,IAAI,CAAC,UA  
U,OAAV,CAAL,C;UAAyB,OAAO,K;;MACtD,OAAO,I;K;8EAGX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB  
;QAAGB,cAAA,SAAhB,M;QAAsB,IAAI,CAAC,UAAU,OAAV,CAAL,C;UAAyB,OAAO,K;;MACtD,OAAO,I;K;  
8EAGX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QAAsB,IAAI,CAAC,UAAU,O  
AAV,CAAL,C;UAAyB,OAAO,K;;MACtD,OAAO,I;K;8EAGX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;Q  
AAGB,cAAA,SAAhB,M;QAAsB,IAAI,CAAC,UAAU,OAAV,CAAL,C;UAAyB,OAAO,K;;MACtD,OAAO,I;K;8E  
AGX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QAAsB,IAAI,CAAC,UAAU,OA  
AV,CAAL,C;UAAyB,OAAO,K;;MACtD,OAAO,I;K;8EAGX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;QA  
AGB,cAAA,SAAhB,M;QAAsB,IAAI,CAAC,UAAU,OAAV,CAAL,C;UAAyB,OAAO,K;;MACtD,OAAO,I;K;8EA

GX,yB;MAAA,oC;MAAA,gC;MAAA,uC;QAMoB,Q;QAaHb,wBAAgB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SA  
AhB,O;UAAsB,IAAI,CAAC,UAAU,oBAAV,CAAL,C;YAAyB,OAAO,K;QACtD,OAAO,I;O;KAPX,C;IAUA,w  
B;MAMI,OAAO,EA5mJA,qBAAQ,CA4mJR,C;K;IAGX,0B;MAMI,OAAO,EA7mJA,qBAAQ,CA6mJR,C;K;IAG  
X,0B;MAMI,OAAO,EA9mJA,qBAAQ,CA8mJR,C;K;IAGX,0B;MAMI,OAAO,EA/mJA,qBAAQ,CA+mJR,C;K;I  
AGX,0B;MAMI,OAAO,EAhnJA,qBAAQ,CagnJR,C;K;IAGX,0B;MAMI,OAAO,EAjnJA,qBAAQ,CAinJR,C;K;I  
AGX,0B;MAMI,OAAO,EAlnJA,qBAAQ,CAknJR,C;K;IAGX,0B;MAMI,OAAO,EAnnJA,qBAAQ,CamnJR,C;K;I  
AGX,0B;MAMI,OAAO,EApnJA,qBAAQ,CAonJR,C;K;8EAGX,gC;MAMoB,Q;MAAhB,wBAAgB,SAAhB,gB;Q  
AAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,I;MACrD,OAAO,K;K;8EAGX,g  
C;MAMoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;U  
AAwB,OAAO,I;MACrD,OAAO,K;K;+EAGX,gC;MAMoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SA  
AhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,I;MACrD,OAAO,K;K;+EAGX,gC;MAMoB,Q;M  
AAhB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,I;  
MACrD,OAAO,K;K;+EAGX,gC;MAMoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,I  
AAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,I;MACrD,OAAO,K;K;+EAGX,gC;MAMoB,Q;MAAhB,wBAAgB,  
SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,I;MACrD,OAAO,  
K;K;+EAGX,gC;MAMoB,Q;MAAhB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OA  
AV,CAAJ,C;UAAwB,OAAO,I;MACrD,OAAO,K;K;+EAGX,gC;MAMoB,Q;MAAhB,wBAAgB,SAAhB,gB;QA  
AgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,I;MACrD,OAAO,K;K;+EAGX,yB  
;MAAA,oC;MAAA,gC;MAAA,uC;QAMoB,Q;QAaHb,wBAAgB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O  
;UAAsB,IAAI,UAAU,oBAAV,CAAJ,C;YAAwB,OAAO,I;QACrD,OAAO,K;O;KAPX,C;gFAUA,qB;MAKI,OA  
AO,gB;K;kFAGX,qB;MAKI,OAAO,gB;K;kFAGX,qB;MAKI,OAAO,gB;K;kFAGX,qB;MAKI,OAAO,gB;K;kFA  
GX,qB;MAKI,OAAO,gB;K;kFAGX,qB;MAKI,OAAO,gB;K;kFAGX,qB;MAKI,OAAO,gB;K;kFAGX,qB;MAKI,  
OAAO,gB;K;kFAGX,qB;MAKI,OAAO,gB;K;kFAGX,gC;MAKoB,Q;MADhB,YAAY,C;MACZ,wBAAgB,SAAh  
B,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,qB;;MAC9C,OAAO,K;K;kFAGX  
,gC;MAKoB,Q;MADhB,YAAY,C;MACZ,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,  
OAAV,CAAJ,C;UAAwB,qB;;MAC9C,OAAO,K;K;mFAGX,gC;MAKoB,Q;MADhB,YAAY,C;MACZ,wBAAgB,  
SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,qB;;MAC9C,OAAO,K;K;  
mFAGX,gC;MAKoB,Q;MADhB,YAAY,C;MACZ,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI  
,UAAU,OAAV,CAAJ,C;UAAwB,qB;;MAC9C,OAAO,K;K;mFAGX,gC;MAKoB,Q;MADhB,YAAY,C;MACZ,w  
BAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,qB;;MAC9C,OAA  
O,K;K;mFAGX,gC;MAKoB,Q;MADhB,YAAY,C;MACZ,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAs  
B,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,qB;;MAC9C,OAAO,K;K;mFAGX,gC;MAKoB,Q;MADhB,YAAY,C;M  
ACZ,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,qB;;MAC9  
C,OAAO,K;K;mFAGX,gC;MAKoB,Q;MADhB,YAAY,C;MACZ,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,  
M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,qB;;MAC9C,OAAO,K;K;mFAGX,yB;MAAA,oC;MAAA,gC;  
MAAA,uC;QAKoB,Q;QADhB,YAAY,C;QACZ,wBAAgB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UAAs  
B,IAAI,UAAU,oBAAV,CAAJ,C;YAAwB,qB;;QAC9C,OAAO,K;O;KANX,C;8EASA,yC;MAUoB,Q;MADhB,kB  
AAkB,O;MACIB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,UAAU,WAAV,EAAuB,OAAv  
B,C;;MACpC,OAAO,W;K;gFAGX,yC;MAUoB,Q;MADhB,kBAAkB,O;MACIB,wBAAgB,SAAhB,gB;QAAgB,c  
AAA,SAAhB,M;QAAsB,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;;MACpC,OAAO,W;K;gFAGX,yC;MAUoB,Q;  
MADhB,kBAAkB,O;MACIB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,UAAU,WAAV,EA  
AuB,OAAvB,C;;MACpC,OAAO,W;K;gFAGX,yC;MAUoB,Q;MADhB,kBAAkB,O;MACIB,wBAAgB,SAAhB,gB  
;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;;MACpC,OAAO,W;K;gFAGX,yC;  
MAUoB,Q;MADhB,kBAAkB,O;MACIB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,UAAU,  
WAAV,EAAuB,OAAvB,C;;MACpC,OAAO,W;K;gFAGX,yC;MAUoB,Q;MADhB,kBAAkB,O;MACIB,wBAAgB  
,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;;MACpC,OAAO,W;K;g  
FAGX,yC;MAUoB,Q;MADhB,kBAAkB,O;MACIB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cA  
Ac,UAAU,WAAV,EAAuB,OAAvB,C;;MACpC,OAAO,W;K;gFAGX,yC;MAUoB,Q;MADhB,kBAAkB,O;MACI

B,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;;MACpC,O  
 AAO,W;K;gFAGX,yB;MAAA,oC;MAAA,gC;MAAA,gD;QAUoB,Q;QADhB,kBAAkB,O;QACIB,wBAAgB,SA  
 AhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UAAsB,cAAc,UAAU,WAAV,EAAuB,oBAAvB,C;;QACpC,OAAO,W  
 ;O;KAXX,C;4FAcA,yC;MAYoB,UAA8B,M;MAF9C,YAAY,C;MACZ,kBAAkB,O;MACIB,wBAAgB,SAAhB,gB  
 ;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,WAAU,cAAV,EAAU,sBAAV,WAAmB,WAAAnB,EAAgC,OAAhC,C;;  
 MACpC,OAAO,W;K;8FAGX,yC;MAYoB,UAA8B,M;MAF9C,YAAY,C;MACZ,kBAAkB,O;MACIB,wBAAgB,S  
 AAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,WAAU,cAAV,EAAU,sBAAV,WAAmB,WAAAnB,EAAgC,O  
 AAhC,C;;MACpC,OAAO,W;K;8FAGX,yC;MAYoB,UAA8B,M;MAF9C,YAAY,C;MACZ,kBAAkB,O;MACIB,w  
 BAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,WAAU,cAAV,EAAU,sBAAV,WAAmB,WAAAnB,E  
 AAgC,OAAhC,C;;MACpC,OAAO,W;K;8FAGX,yC;MAYoB,UAA8B,M;MAF9C,YAAY,C;MACZ,kBAAkB,O;  
 MACIB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,WAAU,cAAV,EAAU,sBAAV,WAAmB,  
 WAAAnB,EAAgC,OAAhC,C;;MACpC,OAAO,W;K;8FAGX,yC;MAYoB,UAA8B,M;MAF9C,YAAY,C;MACZ,kB  
 AAkB,O;MACIB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,WAAU,cAAV,EAAU,sBAAV,  
 WAAmB,WAAAnB,EAAgC,OAAhC,C;;MACpC,OAAO,W;K;8FAGX,yC;MAYoB,UAA8B,M;MAF9C,YAAY,C;  
 MACZ,kBAAkB,O;MACIB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,WAAU,cAAV,EAA  
 U,sBAAV,WAAmB,WAAAnB,EAAgC,OAAhC,C;;MACpC,OAAO,W;K;8FAGX,yC;MAYoB,UAA8B,M;MAF9C,  
 YAAY,C;MACZ,kBAAkB,O;MACIB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,WAAU,cA  
 AV,EAAU,sBAAV,WAAmB,WAAAnB,EAAgC,OAAhC,C;;MACpC,OAAO,W;K;8FAGX,yC;MAYoB,UAA8B,M  
 ;MAF9C,YAAY,C;MACZ,kBAAkB,O;MACIB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,cAAc,W  
 AAU,cAAV,EAAU,sBAAV,WAAmB,WAAAnB,EAAgC,OAAhC,C;;MACpC,OAAO,W;K;8FAGX,yB;MAAA,oC;  
 MAAA,gC;MAAA,gD;QAYoB,UAA8B,M;QAF9C,YAAY,C;QACZ,kBAAkB,O;QACIB,wBAAgB,SAAhB,gB;U  
 AAgB,cAAhB,UAAgB,SAAhB,O;UAAsB,cAAc,WAAU,cAAV,EAAU,sBAAV,WAAmB,WAAAnB,EAAgC,oBA  
 AhC,C;;QACpC,OAAO,W;O;KAbX,C;wFAGBA,yB;MAAA,8D;MAAA,gD;QAYoC,Q;QAHhC,YAAY,wB;QAC  
 Z,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,YAAJ,EAAI,oBAAJ,OAAV,EAAwB,  
 WAAxB,C;;QAEIB,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,8D;MAAA,gD;QAYoC,Q;QAHhC,YAAY,wB;QA  
 CZ,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,YAAJ,EAAI,oBAAJ,OAAV,EAAw  
 B,WAAxB,C;;QAEIB,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,8D;MAAA,gD;QAYoC,Q;QAHhC,YAAY,wB;Q  
 ACZ,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,YAAJ,EAAI,oBAAJ,OAAV,EA  
 AwB,WAAxB,C;;QAEIB,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,8D;MAAA,gD;QAYoC,Q;QAHhC,YAAY,wB  
 ;QACZ,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,YAAJ,EAAI,oBAAJ,OAAV,EA  
 AwB,WAAxB,C;;QAEIB,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,8D;MAAA,gD;QAYoC,Q;QAHhC,YAAY,wB  
 ;QACZ,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,YAAJ,EAAI,oBAAJ,OAAV,EA  
 AwB,WAAxB,C;;QAEIB,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,8D;MAAA,gD;QAYoC,Q;QAHhC,YAA  
 Y,wB;QACZ,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,YAAJ,EAAI,oBAAJ,OAA  
 V,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KAdX,C;0FAiBA,yB;MAAA,8D;MAAA,oC;MAAA,gD;QAYoC,Q;  
 QAHhC,YAAY,wB;QACZ,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,sBAAI,YAAJ,EAA  
 I,oBAAJ,QAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KAdX,C;sGAiBA,yB;MAAA,8D;MAAA,gD;QAUI,Y  
 AAY,wB;QACZ,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KA AV,EAAiB,UAAI,KA  
 AJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAhBX,C;wGAmBA,yB;MAAA,8D;MAAA,gD;QAU  
 I,YAAY,wB;QACZ,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KA AV,EAAiB,UAAI,KA  
 AJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAhBX,C;wGAmBA,yB;MAAA,8D;MAAA,gD;Q  
 AUI,YAAY,wB;QACZ,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KA AV,EAAiB,UAAI,  
 KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAhBX,C;wGAmBA,yB;MAAA,8D;MAAA,g  
 D;QAUI,YAAY,wB;QACZ,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KA AV,EAAiB,UA

AI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAhBX,C;wGAmBA,yB;MAAA,8D;MAAA ,gD;QAUI,YAAY,wB;QACZ,kBAakB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,U AAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAhBX,C;wGAmBA,yB;MAAA,8D;MAA A,gD;QAUI,YAAY,wB;QACZ,kBAakB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB, UAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAhBX,C;wGAmBA,yB;MAAA,8D;MA A,A,gD;QAUI,YAAY,wB;QACZ,kBAakB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAi B,UAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAhBX,C;wGAmBA,yB;MAAA,8D; MAAA,gD;QAUI,YAAY,wB;QACZ,kBAakB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,E AAI,B,UAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAhBX,C;wGAmBA,yB;MAAA,8 D;MAAA,oC;MAAA,gD;QAUI,YAAY,wB;QACZ,kBAakB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UA AU,KAAV,EAAiB,sBAAI,KAAJ,EAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAhBX,C;oFAmBA ,6B;MAIoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QAAsB,OAAO,OAAP,C;;K;sFAG1B,6B ;MAIoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QAAsB,OAAO,OAAP,C;;K;sFAG1B,6B;M AIoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QAAsB,OAAO,OAAP,C;;K;sFAG1B,6B;MAI oB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QAAsB,OAAO,OAAP,C;;K;sFAG1B,6B;MAIoB, Q;MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QAAsB,OAAO,OAAP,C;;K;sFAG1B,6B;MAIoB,Q; MAAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QAAsB,OAAO,OAAP,C;;K;sFAG1B,6B;MAIoB,Q;M AAhB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QAAsB,OAAO,OAAP,C;;K;sFAG1B,6B;MAIoB,Q;MAA hB,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QAAsB,OAAO,OAAP,C;;K;sFAG1B,yB;MAAA,oC;MAAA, gC;MAAA,oC;QAIOB,Q;QAAhB,wBAAGB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UAAsB,OAAO,oBA AP,C;;O;KAJ1B,C;kGAOA,6B;MAOiB,UAAa,M;MAD1B,YAAY,C;MACZ,wBAAa,SAAb,gB;QAAa,WAAA,SA Ab,M;QAAMB,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;K;oGAGvB,6B;MAOiB,UAAa,M;MAD1B,YA AY,C;MACZ,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QAAMB,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB, C;;K;oGAGvB,6B;MAOiB,UAAa,M;MAD1B,YAAY,C;MACZ,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QAA mB,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;K;oGAGvB,6B;MAOiB,UAAa,M;MAD1B,YAAY,C;MAC Z,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QAAMB,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;K;oGAG vB,6B;MAOiB,UAAa,M;MAD1B,YAAY,C;MACZ,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QAAMB,QAAO,c AAP,EAAO,sBAAP,WAAgB,IAAhB,C;;K;oGAGvB,6B;MAOiB,UAAa,M;MAD1B,YAAY,C;MACZ,wBAAa,SA Ab,gB;QAAa,WAAA,SAAb,M;QAAMB,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;K;oGAGvB,6B;MAOi B,UAAa,M;MAD1B,YAAY,C;MACZ,wBAAa,SAAb,gB;QAAa,WAAA,SAAb,M;QAAMB,QAAO,cAAP,EAAO, sBAAP,WAAgB,IAAhB,C;;K;oGAGvB,6B;MAOiB,UAAa,M;MAD1B,YAAY,C;MACZ,wBAAa,SAAb,gB;QAA a,WAAA,SAAb,M;QAAMB,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;K;oGAGvB,yB;MAAA,oC;MAAA ,gC;MAAA,oC;QAoiB,UAAa,M;QAD1B,YAAY,C;QACZ,wBAAa,SAAb,gB;UAAa,WAAb,UAAa,SAAb,O;UA AmB,QAAO,cAAP,EAAO,sBAAP,WAAgB,iBAAhB,C;;O;KAPvB,C;IAUA,wB;MAII,OAAO,oB;K;IAGX,0B;M AII,OAAO,sB;K;IAGX,0B;MAGI,OAAO,sB;K;IAGX,0B;MAGI,OAAO,sB;K;IAGX,0B;MAGI,OAAO,sB;K;IAG X,0B;MAGI,OAAO,sB;K;IAGX,0B;MAGI,OAAO,sB;K;IAGX,0B;MAGI,OAAO,sB;K;IAGX,0B;MAGI,OAAO,s B;K;IAGX,0B;MAGI,OAAO,sB;K;gFAGX,yB;MAsDA,8D;MAtdA,sC;QAGW,sB;;UA0DP,IAhxLO,qBAAQ,CA gxLf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAj B,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eA9DmB,QA8DJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb, M;YACI,QAAQ,UAAK,CAAL,C;YACR,QAJEe,QAIeP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;c ACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QAvEP,yB;O;KAHJ,C;kFAMA,yB;MAuEA,8D;MAvEA,s C;QAGW,sB;;UA2EP,IA/xLO,qBAAQ,CA+xLf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UA Cd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eA/EmB,QA+EJ,CAAS,O AAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QAIfe,QakFP,CAAS,CAAT, C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QAxFP,yB;O;K AHJ,C;kFAMA,yB;MAwFA,8D;MAxFA,sC;QAGW,sB;;UA4FP,IA9yLO,qBAAQ,CA8yLf,C;YAAe,qBAAO,I;Y AAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;Y AAP,uB;WACpB,eAhGmB,QAgGJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,C

AAL,C;YACR,QAnGe,QAmGP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QAzGP,yB;O;KAHJ,C;kFAMA,yB;MAyGA,8D;MAzGA,sC;QAGW,sB;;UA6GP,IA7zLO,qBAAQ,CA6zLf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eAjHmB,QAiHJ,CAAS,OAAT,C;UACf,aAAU,C;AAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QApHe,QAoHP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA1HP,yB;O;KAHJ,C;kFAMA,yB;MA0HA,8D;MA1HA,sC;QAGW,sB;;UA8HP,IA50LO,qBAAQ,CA40Lf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eAllmB,QAkIJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QArIe,QAqIP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA3IP,yB;O;KAHJ,C;kFAMA,yB;MA2IA,8D;MA3IA,sC;QAGW,sB;;UA+IP,IA31LO,qBAAQ,CA21Lf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eAnJmB,QAmJJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QAtJe,QAsJP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA5JP,yB;O;KAHJ,C;kFAMA,yB;MA4JA,8D;MA5JA,sC;QAGW,sB;;UAgKP,IA12LO,qBAAQ,CA02Lf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eApKmB,QAoKJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QAvKe,QAuKP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA7KP,yB;O;KAHJ,C;kFAMA,yB;MA6KA,8D;MA7KA,sC;QAGW,sB;;UAIpL,IAz3LO,qBAAQ,CAY3Lf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eArLmB,QAqLJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QAxLe,QAwLP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA9LP,yB;O;KAHJ,C;kFAMA,yB;MA8LA,8D;MAAA,oC;MA9LA,sC;QAGW,sB;;UAKMP,IAx4LO,qBAAQ,CAw4Lf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eAtMmB,QAsMJ,CAAS,oBAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QAzMe,QAyMP,CAAS,cAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA/MP,yB;O;KAHJ,C;4FAMA,yB;MAAA,8D;MAAA,sC;QAOI,IAhxLO,qBAAQ,CagxLf,C;UAAe,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KApBX,C;8FAuBA,yB;MAAA,8D;MAAA,sC;QAOI,IA9yLO,qBAAQ,CA8yLf,C;UAAe,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KApBX,C;8FAuBA,yB;MAAA,8D;MAAA,sC;QAOI,IA7zLO,qBAAQ,CA6zLf,C;UAAe,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KApBX,C;8FAuBA,yB;MAAA,8D;MAAA,sC;QAOI,IA50LO,qBAAQ,CA40Lf,C;UAAe,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KApBX,C;8FAuBA,yB;MAAA,8D;MAAA,sC;QAOI,IA31LO,qBAAQ,CA21Lf,C;U

Ae,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAA  
oB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;  
UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGn  
B,OAAO,O;O;KApBX,C;8FAuBA,yB;MAAA,8D;MAAA,sC;QAOI,IA12LO,qBAAQ,CA02Lf,C;UAAe,OAAO,I;  
QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;  
QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAA  
Q,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;  
O;KApBX,C;8FAuBA,yB;MAAA,8D;MAAA,sC;QAOI,IAz3LO,qBAAQ,CAY3Lf,C;UAAe,OAAO,I;QACtB,cAA  
c,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eA  
Ae,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,C  
AAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KApBX,  
C;8FAuBA,yB;MAAA,8D;MAAA,oC;MAAA,sC;QAOI,IAx4LO,qBAAQ,CAw4Lf,C;UAAe,OAAO,I;QACtB,cA  
Ac,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,e  
AAe,SAAS,oBAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,  
cAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KApBX,  
C;gFAuBA,yB;MAAA,sE;MAAA,8D;MkBhnbA,iB;MIBgnbA,sC;QAeiB,Q;QAFb,IAr+LO,qBAAQ,CAq+Lf,C;U  
AAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,S  
AAS,UAAK,CAAL,CAAT,C;UACR,WkBznbg,MAAO,KlBznbO,QkBznbP,ElBznbiB,CkBznbjB,C;;QIB2nbd,OA  
AO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkBtobA,iB;MIBsobA,sC;QAeiB,Q;QAFb,IAr/LO,qBAA  
Q,CAm/Lf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;  
UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB/obG,MAAO,KIB+obO,QkB/obP,ElB+obiB,CkB/objB,  
C;;QIBipbd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkB5pbA,iB;MIB4pbA,sC;QAeiB,Q;QAF  
b,IAjgMO,qBAAQ,CAigMf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,  
aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBBrqB,MAAO,KIBqqbO,QkBBrqP,ElB  
qqbiB,CkBBrqbjB,C;;QIBuqbd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkBlrbA,iB;MIBkbrA,s  
C;QAeiB,Q;QAFb,IA/gMO,qBAAQ,CA+gMf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;  
QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB3rbG,MAAO,KIB2r  
bO,QkB3rbP,ElB2rbiB,CkB3rbjB,C;;QIB6rbd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkBxsb  
A,iB;MIBwsbA,sC;QAeiB,Q;QAFb,IA7hMO,qBAAQ,CA6hMf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,  
CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBjtb  
G,MAAO,KIBitbO,QkBjtbP,ElBitbiB,CkBjtbjB,C;;QIBmtbd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAA  
A,8D;MkB9tbA,iB;MIB8tbA,sC;QAeiB,Q;QAFb,IA3iMO,qBAAQ,CA2iMf,C;UAAe,MAAM,6B;QACrB,eAAe,S  
AAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;U  
ACR,WkBvubG,MAAO,KIBuubO,QkBvubP,ElBuubiB,CkBvubjB,C;;QIByubd,OAAO,Q;O;KAnBX,C;kFAsBA,y  
B;MAAA,sE;MAAA,8D;MkBpvaA,iB;MIBovbA,sC;QAeiB,Q;QAFb,IAzjMO,qBAAQ,CAYjMf,C;UAAe,MAAM,  
6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,  
CAAL,CAAT,C;UACR,WkB7vbG,MAAO,KIB6vbO,QkB7vbP,ElB6vbiB,CkB7vbjB,C;;QIB+vbd,OAAO,Q;O;KA  
nBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkB1wbA,iB;MIB0wbA,sC;QAeiB,Q;QAFb,IAvkMO,qBAAQ,CAuk  
Mf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,  
QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBnxbG,MAAO,KIBmxbO,QkBnxbP,ElBmxbiB,CkBnxbjB,C;;QI  
Bqxbd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,oC;MAAA,8D;MkBhybA,iB;MIBgybA,sC;QAeiB,  
Q;QAFb,IArlMO,qBAAQ,CAqlMf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;  
QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,WkBzybG,MAAO,KIByybO,QkBz  
ybP,ElByybiB,CkBzybjB,C;;QIB2ybd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkBj0bA,iB;MI  
Bi0bA,sC;QAeiB,Q;QAFb,IA3qMO,qBAAQ,CA2qMf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,C  
AAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB10bG,MAA  
O,KIB00bO,QkB10bP,ElB00biB,CkB10bjB,C;;QIB40bd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8  
D;MkBv1bA,iB;MIBu1bA,sC;QAeiB,Q;QAFb,IAzrMO,qBAAQ,CAYrMf,C;UAAe,MAAM,6B;QACrB,eAAe,SA

AS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBh2bG,MAAO,KIBg2bO,QkBh2bP,ElBg2biB,CkBh2bjB,C;;QlBk2bd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MkB72bA,iB;MIB62bA,sC;QAeiB,Q;QAFb,IAvsMO,qBAAQ,CAusMf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBt3bG,MAAO,KIBs3bO,QkBt3bP,ElBs3biB,CkBt3bjB,C;;QlBw3bd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MkBn4bA,iB;MIBm4bA,sC;QAeiB,Q;QAFb,IArtMO,qBAAQ,CAqtMf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB54bG,MAAO,KIB44bO,QkB54bP,ElB44biB,CkB54bjB,C;;QlB84bd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MkBz5bA,iB;MIBy5bA,sC;QAeiB,Q;QAFb,IANuMO,qBAAQ,CAmuMf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB16bG,MAAO,KIBk6bO,QkB16bP,ElBk6biB,CkB16bjB,C;;QlBo6bd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MkB/6bA,iB;MIB+6bA,sC;QAeiB,Q;QAFb,IAjvMO,qBAAQ,CAivMf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBx7bG,MAAO,KIBw7bO,QkBx7bP,ElBw7biB,CkBx7bjB,C;;QlB07bd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MkB8bA,iB;MIBq8bA,sC;QAeiB,Q;QAFb,IA/vMO,qBAAQ,CA+vMf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB98bG,MAAO,KIB88bO,QkB98bP,ElB88biB,CkB98bjB,C;;QlB9bd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MkB39bA,iB;MIB29bA,sC;QAeiB,Q;QAFb,IA7wMO,qBAAQ,CA6wMf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBp+bG,MAAO,KIBo+bO,QkBp+bP,ElBo+biB,CkBp+bjB,C;;QlBs+bd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,oC;MAAA,8D;MkBj/bA,iB;MIBi/bA,sC;QAeiB,Q;QAFb,IA3xMO,qBAAQ,CA2xMf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,WkB1/bG,MAAO,KIB0/bO,QkB1/bP,ElB0/biB,CkB1/bjB,C;;QlB4/bd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IA/2MO,qBAAQ,CA+2Mf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IA73MO,qBAAQ,CA63Mf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IA34MO,qBAAQ,CA24Mf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IAz5MO,qBAAQ,CAy5Mf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IAv6MO,qBAAQ,CAu6Mf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IAr7MO,qBAAQ,CAq7Mf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IAN8MO,qBAAQ,CAn8Mf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IAj9MO,qBAAQ,CAi9Mf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,oC;MAAA,8D;MAAA,

sC;QAaiB,Q;QAFb,IA/9MO,qBAAQ,CA+9Mf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,IAAI,2BAAW,CAAX,KAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;4FAsBA,yB;MAAA,8D;MkBlscA,iB;MIBkscA,sC;QAaiB,Q;QAFb,IArjNO,qBAAQ,CAqjNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBzscG,MAAO,KIByscO,QkBzscP,ElBysciB,CkBzscjB,C;;QIB2scd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkBttcA,iB;MIBstcA,sC;QAaiB,Q;QAFb,IAjkNO,qBAAQ,CAikNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB7tcG,MAAO,KIB6tcO,QkB7tcP,ElB6tciB,CkB7tcjB,C;;QIB+tcD,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkB1ucA,iB;MIB0ucA,sC;QAaiB,Q;QAFb,IA7kNO,qBAAQ,CA6kNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBjvcG,MAAO,KIBivcO,QkBjvcP,ElBivciB,CkBjvcjB,C;;QIBmvcd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkB9vcA,iB;MIB8vcA,sC;QAaiB,Q;QAFb,IAzInO,qBAAQ,CAylNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBrcG,MAAO,KIBqwcO,QkBrcP,ElBqwciB,CkBrcjB,C;;QIBuwcd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkB1xcA,iB;MIBkxcA,sC;QAaiB,Q;QAFb,IArmNO,qBAAQ,CAqmNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBzxcG,MAAO,KIByxcO,QkBzxcP,ElByxciB,CkBzxcjB,C;;QIB2xcd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkBtycA,iB;MIBsycA,sC;QAaiB,Q;QAFb,IAjnNO,qBAAQ,CAinNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB7ycG,MAAO,KIB6ycO,QkB7ycP,ElB6yciB,CkB7ycjB,C;;QIB+ycd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkB1zcA,iB;MIB0zcA,sC;QAaiB,Q;QAFb,IA7nNO,qBAAQ,CA6nNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBj0cG,MAAO,KIBi0cO,QkBj0cP,ElBi0ciB,CkBj0cjB,C;;QIBm0cd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkB90cA,iB;MIB80cA,sC;QAaiB,Q;QAFb,IAzoNO,qBAAQ,CAyoNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBrlcG,MAAO,KIBq1cO,QkBrlcP,ElBq1ciB,CkBrlcjB,C;;QIBu1cd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkB12cA,iB;MIBk2cA,sC;QAaiB,Q;QAFb,IArpNO,qBAAQ,CAqpNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,WkBz2cG,MAAO,KIBy2cO,QkBz2cP,ElBy2ciB,CkBz2cjB,C;;QIB22cd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkBj4cA,iB;MIBi4cA,sC;QAaiB,Q;QAFb,IAzuNO,qBAAQ,CAyuNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBx4cG,MAAO,KIBw4cO,QkBx4cP,ElBw4ciB,CkBx4cjB,C;;QIB04cd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkB5cA,iB;MIBq5cA,sC;QAaiB,Q;QAFb,IArvNO,qBAAQ,CAqvNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB55cG,MAAO,KIB45cO,QkB55cP,ElB45ciB,CkB55cjB,C;;QIB85cd,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MkBz6cA,iB;MIBy6cA,sC;QAaiB,Q;QAFb,IAjwNO,qBAAQ,CAiwNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBh7cG,MAAO,KIBg7cO,QkBh7cP,ElBg7ciB,CkBh7cjB,C;;QIBk7cd,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MkB77cA,iB;MIB67cA,sC;QAaiB,Q;QAFb,IA7wNO,qBAAQ,CA6wNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBp8cG,MAAO,KIBo8cO,QkBp8cP,ElBo8ciB,CkBp8cjB,C;;QIBs8cd,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MkBj9cA,iB;MIBi9cA,sC;QAaiB,Q;QAFb,IAzxNO,qBAAQ,CAyxNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBx9cG,MAAO,KIBw9cO,QkBx9cP,ElBw9ciB,CkBx9cjB,C;;QIB09cd,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MkBrcA,iB;MIBq+cA,sC;QAaiB,Q;QAFb,IARYNO,qBAAQ,CAqyNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB5+cG,MAAO,KIB4+cO,QkB5+cP,ElB4+ciB,CkB5+cjB,C;;QIB8+cd,OAAO,Q;O;KAjBX,C;+FA

oBA,yB;MAAA,8D;MkBz/cA,iB;MIBy/cA,sC;QAaiB,Q;QAFb,IAjzNO,qBAAQ,CAizNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBhgdG,MAAO,KIBggdO,QkBhgdP,ElBggdiB,CkBghdjB,C;;QIBkgdd,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MkB7gdA,iB;MIB6gdA,sC;QAaiB,Q;QAFb,IA7zNO,qBAAQ,CA6zNf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBphdG,MAAO,KIBohdO,QkBphdP,ElBohdiB,CkBphdjB,C;;QIBshdd,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,oC;MAAA,8D;MkBjidA,iB;MIBiidA,sC;QAaiB,Q;QAFb,IAz0NO,qBAAQ,CAy0Nf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,WkBxidG,MAAO,KIBwidO,QkBxidP,ElBwidiB,CkBxidjB,C;;QIB0idd,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IA35NO,qBAAQ,CA25Nf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IAv6NO,qBAAQ,CAu6Nf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IA7NO,qBAAQ,CAm7Nf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IA/7NO,qBAAQ,CA+7Nf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IAv9NO,qBAAQ,CAu9Nf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IAN+NO,qBAAQ,CAm+Nf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IA+/NO,qBAAQ,CA++Nf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,oC;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IA3/NO,qBAAQ,CA2/Nf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;wFAoBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IAjloo,qBAAQ,CAiiof,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IA/100,qBAAQ,CA+10f,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IA7mOO,qBAAQ,CA6mOf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IA3nOO,qBAAQ,CA2nOf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IAzoOO,qBA

AQ,CAyoOf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IAvpOO,qBAAQ,CAupOf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IArqOO,qBAAQ,CAqqOf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IAnrOO,qBAAQ,CAMrOf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IAjsOO,qBAAQ,CAisOf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;oGAsBA,yB;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IArxOO,qBAAQ,CAqxOf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IAjyOO,qBAAQ,CAiyOf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IA7yOO,qBAAQ,CA6yOf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IAzzOO,qBAAQ,CAyzOf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IAr0OO,qBAAQ,CAq0Of,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IAj1OO,qBAAQ,CAi1Of,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IAz2OO,qBAAQ,CAy2Of,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IAz3OO,qBAAQ,CAq3Of,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;IAoBA,8B;MASiB,Q;MAFb,IAv8OO,qBAAQ,CAu8Of,C;QAAe,OAAO,I;MACtB,UAAU,UAAK,CAAL,C;MACG,+B;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,MkB3leG,MAAO,KIB2leE,GkB3leF,EIB2leO,CkB3leP,C;;MIB6led,OAAO,G;K;IAGX,gC;MASiB,Q;MAFb,IAv9OO,qBAAQ,CAu9Of,C;QAAe,OAAO,I;MACtB,UAAU,UAAK,CAAL,C;MACG,+B;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,MkBtne



A,sC;QAGW,sB;;UA0DP,IAAn4PO,qBAAQ,CAM4Pf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eA9DmB,QA8DJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QAjEe,QAiEP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QAvEP,yB;O;KAHJ,C;kFAMA,yB;MAuEA,8D;MAvEA,sC;QAGW,sB;;UA2EP,IAI5PO,qBAAQ,Cak5Pf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eA/EmB,QA+EJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QAIfe,QakFP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QAxFP,yB;O;KAHJ,C;kFAMA,yB;MAwFA,8D;MAxFA,sC;QAGW,sB;;UA4FP,IAj6PO,qBAAQ,CAi6Pf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eAhGmB,QAgGJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QAnGe,QAmGP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QAzGP,yB;O;KAHJ,C;kFAMA,yB;MAyGA,8D;MAzGA,sC;QAGW,sB;;UA6GP,IAh7PO,qBAAQ,Cag7Pf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eAjHmB,QaiHJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QApHe,QAoHP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA1HP,yB;O;KAHJ,C;kFAMA,yB;MA0HA,8D;MA1HA,sC;QAGW,sB;;UA8HP,IA/7PO,qBAAQ,CA+7Pf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eAllmB,QakIJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QArIe,QaqIP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA3IP,yB;O;KAHJ,C;kFAMA,yB;MA2IA,8D;MA3IA,sC;QAGW,sB;;UA+IP,IA98PO,qBAAQ,CA88Pf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eAnJmB,QAmJJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QatJe,QAsJP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA5JP,yB;O;KAHJ,C;kFAMA,yB;MA4JA,8D;MA5JA,sC;QAGW,sB;;UAgKP,IA79PO,qBAAQ,CA69Pf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eApKmB,QAoKJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QAvKe,QAuKP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA7KP,yB;O;KAHJ,C;kFAMA,yB;MA6KA,8D;MA7KA,sC;QAGW,sB;;UaiLP,IA5+PO,qBAAQ,CA4+Pf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eArLmB,QAqLJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QAxLe,QAwLP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA9LP,yB;O;KAHJ,C;kFAMA,yB;MA8LA,8D;MAAAA,oC;MA9LA,sC;QAGW,sB;;UakMP,IA3/PO,qBAAQ,CA2/Pf,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,UAAK,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eAtMmB,QAsMJ,CAAS,oBAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,UAAK,CAAL,C;YACR,QAzMe,QAyMP,CAAS,cAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA/MP,yB;O;KAHJ,C;4FAMA,yB;MAAAA,8D;MAAAA,sC;QAOI,IAAn4PO,qBAAQ,CAM4Pf,C;UAAe,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KApBX,C;8FAuBA,yB;MAAAA,8D;MAAAA,sC;QAOI,IAI5PO,qBAAQ,Cak5Pf,C;UAAe,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KApBX,C;8FAuBA,yB;MAAAA,8D;MAAAA,sC;QAOI,IAj6PO,qBAAQ,CAi6Pf,C;UAAe,OAAO,I;QACtB,cAAc,UAA

K,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;KApBX,C;8FAuBA,yB;MAAA,8D;MAAA,sC;QAOI,IAh7PO,qBAAQ,CAg7Pf,C;UAAe,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;KApBX,C;8FAuBA,yB;MAAA,8D;MAAA,sC;QAOI,IA7PO,qBAAQ,CA+7Pf,C;UAAe,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;KApBX,C;8FAuBA,yB;MAAA,8D;MAAA,sC;QAOI,IA98PO,qBAAQ,CA88Pf,C;UAAe,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;KApBX,C;8FAuBA,yB;MAAA,8D;MAAA,sC;QAOI,IA79PO,qBAAQ,CA69Pf,C;UAAe,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;KApBX,C;8FAuBA,yB;MAAA,8D;MAAA,sC;QAOI,IA5+PO,qBAAQ,CA4+Pf,C;UAAe,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;KApBX,C;8FAuBA,yB;MAAA,8D;MAAA,oC;MAAA,sC;QAOI,IA3/PO,qBAAQ,CA2/Pf,C;UAAe,OAAO,I;QACtB,cAAc,UAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,oBAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,UAAK,CAAL,C;UACR,QAAQ,SAAS,cAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;KApBX,C;gFAuBA,yB;MAAA,sE;MAAA,8D;MkB/gfA,iB;MIB+gfA,sC;QAeiB,Q;QAFb,IAxlQO,qBAAQ,CAwlQf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBxhfG,MAAO,KIBwhfO,QkBxhfp,EIBwhfiB,CkBxhfjB,C;;QIB0hfd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkBBrifA,iB;MIBqifA,sC;QAeiB,Q;QAFb,IAtmQO,qBAAQ,CAsmQf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB9ifG,MAAO,KIB8ifO,QkB9ifP,EIB8ifiB,CkB9ifjB,C;;QIBgjfD,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkB3jfA,iB;MIB2jfA,sC;QAeiB,Q;QAFb,IApnQO,qBAAQ,CAonQf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBpkfG,MAAO,KIBokfO,QkBpkip,EIBokfiB,CkBpkfjB,C;;QIBskfd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkBjlfA,iB;MIBilfA,sC;QAeiB,Q;QAFb,IAloQO,qBAAQ,CAkoQf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB1lfG,MAAO,KIB0lfO,QkB1lfP,EIB0lfiB,CkB1lfjB,C;;QIB4lfd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkBvmfA,iB;MIBumfA,sC;QAeiB,Q;QAFb,IAhpQO,qBAAQ,CAGpQf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBhfnG,MAAO,KIBgnfO,QkBhfnP,EIBgnfiB,CkBhfnfjB,C;;QIBknfd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkB7nfA,iB;MIB6nfA,sC;QAeiB,Q;QAFb,IA9pQO,qBAAQ,CA8pQf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBtofG,MAAO,KIBsofO,QkBtofp,EIBsofiB,CkBtofjB,C;;QIBwofd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkBnfpA,iB;MIBmpfA,sC;QAeiB,Q;QAFb,IA5qQO,qBAAQ,CA4qQf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB5pfG,MAAO,KIB4pfO,QkB5pfP,EIB4pfiB,CkB5pfjB,C;;QI

B8pfd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkBzqfA,iB;MlByqfA,sC;QAeiB,Q;QAFb,IA1r  
QO,qBAAQ,CA0rQf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,  
CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBlrfG,MAAO,KlBkrfO,QkBlrfP,ElBkrfiB,CkB  
lrfjB,C;;QlBorfd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,oC;MAAA,8D;MkB/rfA,iB;MlB+rfA,sC;  
QAeiB,Q;QAFb,IAxsQO,qBAAQ,CAwsQf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;Q  
ACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,WkBxsfG,MAAO,KlBwsf  
O,QkBxsfP,ElBwsfiB,CkBxsfjB,C;;QlB0sfd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAAA,8D;MkBhufA  
,iB;MlBguF,a,sC;QAeiB,Q;QAFb,IA9xQO,qBAAQ,CA8xQf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,C  
AAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBzufG,  
MAAO,KlByufO,QkBzufP,ElByufiB,CkBzufjB,C;;QlB2ufd,OAAO,Q;O;KAnBX,C;kFAsBA,yB;MAAA,sE;MAA  
A,8D;MkBtVfA,iB;MlBsvfA,sC;QAeiB,Q;QAFb,IA5yQO,qBAAQ,CA4yQf,C;UAAe,MAAM,6B;QACrB,eAAe,S  
AAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;U  
ACR,WkB/vfG,MAAO,KlB+vfO,QkB/vfP,ElB+vfI,B,CkB/vfjB,C;;QlBiwfd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;  
MAAA,sE;MAAA,8D;MkB5wfA,iB;MlB4wfA,sC;QAeiB,Q;QAFb,IA1zQO,qBAAQ,CA0zQf,C;UAAe,MAAM,6  
B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,C  
AAL,CAAT,C;UACR,WkBxrfG,MAAO,KlBqxfO,QkBxrfP,ElBqxfiB,CkBxrfjB,C;;QlBuxfd,OAAO,Q;O;KAnBX,  
C;mFAsBA,yB;MAAA,sE;MAAA,8D;MkBlyfA,iB;MlBkyfA,sC;QAeiB,Q;QAFb,IAx0QO,qBAAQ,CAw0Qf,C;U  
AAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,S  
AAS,UAAK,CAAL,CAAT,C;UACR,WkB3yfG,MAAO,KlB2yfO,QkB3yfP,ElB2yfiB,CkB3yfiB,C;;QlB6yfd,OAA  
O,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MkBzxfA,iB;MlBwzfA,sC;QAeiB,Q;QAFb,IA1zQO,qBAA  
Q,CAs1Qf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;  
UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBj0fG,MAAO,KlBi0fO,QkBj0fP,ElBi0fiB,CkBj0fiB,C;;Q  
lBm0fd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MkB90fA,iB;MlB80fA,sC;QAeiB,Q;QAFb,I  
Ap2QO,qBAAQ,CAo2Qf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aA  
AU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBv1fG,MAAO,KlBu1fO,QkBv1fP,ElBu1fi  
B,CkBv1fiB,C;;QlBy1fd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MkBp2fA,iB;MlBo2fA,sC;Q  
AeiB,Q;QAFb,IA13QO,qBAAQ,CAk3Qf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QAC  
F,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB72fG,MAAO,KlB62fO,Q  
kB72fP,ElB62fiB,CkB72fiB,C;;QlB+2fd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MkB13fA,iB  
;MlB03fA,sC;QAeiB,Q;QAFb,IAh4QO,qBAAQ,CAg4Qf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAA  
L,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBn4fG,M  
AAO,KlBm4fO,QkBn4fP,ElBm4fiB,CkBn4fiB,C;;QlBq4fd,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAA  
A,oC;MAAA,8D;MkBh5fA,iB;MlBg5fA,sC;QAeiB,Q;QAFb,IA94QO,qBAAQ,CA84Qf,C;UAAe,MAAM,6B;QA  
CrB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAA  
L,EAAT,C;UACR,WkBz5fG,MAAO,KlBy5fO,QkBz5fP,ElBy5fiB,CkBz5fiB,C;;QlB25fd,OAAO,Q;O;KAnBX,C;  
mFAsBA,yB;MAAA,sE;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IAI+QO,qBAAQ,CAk+Qf,C;UAAe,MAAM,6B;Q  
ACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAA  
L,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,y  
B;MAAA,sE;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IAh/QO,qBAAQ,CAg/Qf,C;UAAe,MAAM,6B;QACrB,eAA  
e,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;  
UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE  
;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IA9/QO,qBAAQ,CA8/Qf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAA  
K,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,  
2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;  
MAAA,sC;QAaiB,Q;QAFb,IA5gRO,qBAAQ,CA4gRf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,C  
AAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,C  
AAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MAAA,sC;  
QAaiB,Q;QAFb,IA1hRO,qBAAQ,CA0hRf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QA

CF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IAxiRO,qBAAQ,CAwiRf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IAIjRO,qBAAQ,CAwjRf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OA AO,Q;O;KAnBX,C;mFAsBA,yB;MAAA,sE;MAAA,oC;MAAA,8D;MAAA,sC;QAaiB,Q;QAFb,IAIIRO,qBAAQ,CAkIRf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OA AO,Q;O;KAnBX,C;4FAsBA,yB;MAAA,8D;MkBjmgBA,iB;MlBimgBA,sC;QAaiB,Q;QAFb,IAxqRO,qBAAQ,CA wqRf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,Q AAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBxmgBG,MAAO,KlBwmgBO,QkBxmgBP,ElBwmgBiB,Ckxmg BjB,C;;QIB0mgBd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkBrngBA,iB;MlBqngBA,sC;QAaiB,Q;QAFb ,IAprRO,qBAAQ,CAorRf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAA U,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB5ngBG,MAAO,KlB4ngBO,QkB5ngBP,ElB 4ngBiB,CkB5ngBjB,C;;QIB8ngBd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkBzogBA,iB;MlByogBA,sC; QAaiB,Q;QAFb,IAhsRO,qBAAQ,CAgsRf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF, +B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBhpgBG,MAAO,KlBpgpBO, QkBhpgBP,ElBpgpBiB,CkBhpgBjB,C;;QIBkpgBd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkB7pgBA,iB ;MlB6pgBA,sC;QAaiB,Q;QAFb,IA5sRO,qBAAQ,CA4sRf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL, CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBppgBG,M AAO,KlBoqgBO,QkBppgBP,ElBoqgBiB,CkBppgBjB,C;;QIBsqgBd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8 D;MkBjrgBA,iB;MlBirgBA,sC;QAaiB,Q;QAFb,IAxtRO,qBAAQ,CAwtRf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS, UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR, WkBxrgBG,MAAO,KlBwrgBO,QkBxrgBP,ElBwrgBiB,CkBxrgBjB,C;;QIB0rgBd,OAAO,Q;O;KAjBX,C;8FAoBA ,yB;MAAA,8D;MkBrsqBA,iB;MlBqsgBA,sC;QAaiB,Q;QAFb,IApuRO,qBAAQ,CAouRf,C;UAAe,OAAO,I;QACt B,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,C AAT,C;UACR,WkB5sgBG,MAAO,KlB4sgBO,QkB5sgBP,ElB4sgBiB,CkB5sgBjB,C;;QIB8sgBd,OAAO,Q;O;KAj BX,C;8FAoBA,yB;MAAA,8D;MkBzrgBA,iB;MlByrgBA,sC;QAaiB,Q;QAFb,IAhvRO,qBAAQ,CAgvRf,C;UAAe, OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,U AAK,CAAL,CAAT,C;UACR,WkBhugBG,MAAO,KlBgugBO,QkBhugBP,ElBgugBiB,CkBhugBjB,C;;QIBkugBd, OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkB7ugBA,iB;MlB6ugBA,sC;QAaiB,Q;QAFb,IA5vRO,qBAAQ, CA4vRf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UAC I,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBpvgBG,MAAO,KlBovgBO,QkBpvgBP,ElBovgBiB,CkBpvgB jB,C;;QIBsvgBd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,oC;MAAA,8D;MkBjwgBA,iB;MlBiwgBA,sC;QAai B,Q;QAFb,IAxwRO,qBAAQ,CAwwRf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+ B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,WkBxwgBG,MAAO,KlBwwgBO ,QkBxwgBP,ElBwwgBiB,CkBxwgBjB,C;;QIB0wgBd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAAA,8D;MkBhygBA ,iB;MlBgygBA,sC;QAaiB,Q;QAFb,IA51RO,qBAAQ,CA41Rf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CA AL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBvygBG ,MAAO,KlBuygBO,QkBvygBP,ElBuygBiB,CkBvygBjB,C;;QIByygBd,OAAO,Q;O;KAjBX,C;8FAoBA,yB;MAA A,8D;MkBpzgBA,iB;MlBozgBA,sC;QAaiB,Q;QAFb,IAx2RO,qBAAQ,CAw2Rf,C;UAAe,OAAO,I;QACtB,eAAe, SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;U ACR,WkB3zgBG,MAAO,KlB2zgBO,QkB3zgBP,ElB2zgBiB,CkB3zgBjB,C;;QIB6zgBd,OAAO,Q;O;KAjBX,C;+F

AoBA,yB;MAAA,8D;MkBx0gBA,iB;MIBw0gBA,sC;QAaiB,Q;QAFb,IAp3RO,qBAAQ,CAo3Rf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,C AAL,CAAT,C;UACR,WkB/0gBG,MAAO,KIB+0gBO,QkB/0gBP,ElB+0gBiB,CkB/0gBjB,C,;QIBi1gBd,OAAO,Q; O;KAjBX,C;+FAoBA,yB;MAAA,8D;MkB51gBA,iB;MIB41gBA,sC;QAaiB,Q;QAFb,IAH4RO,qBAAQ,CAG4Rf,C ;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,S AAS,UAAK,CAAL,CAAT,C;UACR,WkBn2gBG,MAAO,KIBm2gBO,QkBn2gBP,ElBm2gBiB,CkBn2gBjB,C,;QI Bq2gBd,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MkBh3gBA,iB;MIBg3gBA,sC;QAaiB,Q;QAFb,IA54RO, qBAAQ,CA44Rf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV, iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkBv3gBG,MAAO,KIBu3gBO,QkBv3gBP,ElBu3gBiB,C kBv3gBjB,C,;QlBy3gBd,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MkBp4gBA,iB;MIBo4gBA,sC;QAaiB,Q ;QAFb,IAx5RO,qBAAQ,CAw5Rf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QA Ab,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB34gBG,MAAO,KIB24gBO,QkB34 gBP,ElB24gBiB,CkB34gBjB,C,;QIB64gBd,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MkBx5gBA,iB;MIBw 5gBA,sC;QAaiB,Q;QAFb,IAp6RO,qBAAQ,CAo6Rf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAA T,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,WkB/5gBG,MAAO, KIB+5gBO,QkB/5gBP,ElB+5gBiB,CkB/5gBjB,C,;QlBi6gBd,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MkB 56gBA,iB;MIB46gBA,sC;QAaiB,Q;QAFb,IAh7RO,qBAAQ,CAG7Rf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UA AK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,Wk Bn7gBG,MAAO,KIBm7gBO,QkBn7gBP,ElBm7gBiB,CkBn7gBjB,C,;QlBq7gBd,OAAO,Q;O;KAjBX,C;+FAoBA, yB;MAAA,oC;MAAA,8D;MkBh8gBA,iB;MIBg8gBA,sC;QAaiB,Q;QAFb,IA57RO,qBAAQ,CA47Rf,C;UAAe,OA AO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBA AK,CAAL,EAAT,C;UACR,WkBv8gBG,MAAO,KIBu8gBO,QkBv8gBP,ElBu8gBiB,CkBv8gBjB,C,;QlBy8gBd,O AAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IA9gSO,qBAAQ,CA8gSf,C;UAAe,O AAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UA AK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C,;QAGnB,OAAO,Q;O;KAjBX,C;+F AoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IA1hSO,qBAAQ,CA0hSf,C;UAAe,OAAO,I;QACtB,eAAe,SA AS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UA CR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C,;QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;M AAA,sC;QAWiB,Q;QAFb,IAtiSO,qBAAQ,CAsiSf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT, C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX, KAAJ,C;YACI,WAAW,C,;QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb, IA1jSO,qBAAQ,CAkjSf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU, CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C,; QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IA9jSO,qBAAQ,CA8jSf,C; UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,S AAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C,;QAGnB,OAAO,Q;O;KAj BX,C;+FAoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IA1kSO,qBAAQ,CA0kSf,C;UAAe,OAAO,I;QACtB,e AAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAA T,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C,;QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAA A,8D;MAAA,sC;QAWiB,Q;QAFb,IAtlSO,qBAAQ,CAslSf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL, CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW, CAAX,KAAJ,C;YACI,WAAW,C,;QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,8D;MAAA,sC;QAWiB,Q ;QAFb,IAImSO,qBAAQ,CAkmSf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QA Ab,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI, WAAW,C,;QAGnB,OAAO,Q;O;KAjBX,C;+FAoBA,yB;MAAA,oC;MAAA,8D;MAAA,sC;QAWiB,Q;QAFb,IA9 mSO,qBAAQ,CA8mSf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU, CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C,; QAGnB,OAAO,Q;O;KAjBX,C;wFAoBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IApsSO,qBAAQ,

CAosSf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;U  
ACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,  
CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAai  
B,Q;QAFb,IAItSO,qBAAQ,CaktSf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;  
QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB  
,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MA  
AA,8D;MAAA,kD;QAaiB,Q;QAFb,IAhuSO,qBAAQ,CaguSf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,  
CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,U  
AAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,  
C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IA9uSO,qBAAQ,CA8uSf,C;UAAe,MAAM,6B;  
QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CA  
AL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;Q  
AGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IA5vSO,qBAAQ,C  
A4vSf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UA  
CI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,C  
AAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB  
,Q;QAFb,IA1wSO,qBAAQ,CA0wSf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B  
;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAak  
B,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;M  
AAA,8D;MAAA,kD;QAaiB,Q;QAFb,IAxxSO,qBAAQ,CawxSf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,UAA  
K,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,  
UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX  
,C;0FAsBA,yB;MAAA,sE;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IAtySO,qBAAQ,CAsySf,C;UAAe,MAAM,6B;  
QACrB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CA  
AL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;Q  
AGnB,OAAO,Q;O;KAnBX,C;0FAsBA,yB;MAAA,sE;MAAA,oC;MAAA,8D;MAAA,kD;QAaiB,Q;QAFb,IApzS  
O,qBAAQ,CAozSf,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,C  
AAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAA  
X,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;oGAsBA,yB;MAAA,8D;MAAA,kD;QA  
WiB,Q;QAFb,IAx4SO,qBAAQ,CAw4Sf,C;UAAe,OAAO,I;QActB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+  
B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAa  
kB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,8D;  
MAAA,kD;QAWiB,Q;QAFb,IAp5SO,qBAAQ,CAo5Sf,C;UAAe,OAAO,I;QActB,eAAe,SAAS,UAAK,CAAL,CA  
AT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAA  
Q,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,y  
B;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IAh6SO,qBAAQ,CAG6Sf,C;UAAe,OAAO,I;QActB,eAAe,SAAS,UA  
AK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAA  
I,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjB  
X,C;sGAoBA,yB;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IA56SO,qBAAQ,CA46Sf,C;UAAe,OAAO,I;QActB,e  
AAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAA  
T,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,O  
AAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IAx7SO,qBAAQ,CAw7Sf,C;UAAe,O  
AAO,I;QActB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UA  
AK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAA  
W,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IAp8SO,qBAAQ,CA  
o8Sf,C;UAAe,OAAO,I;QActB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,Q  
AAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC  
,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,8D;MAAA,kD;QAWiB,Q;QAFb,IAh9S

O,qBAAQ,CAg9Sf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QAAb,aAAU,CAA V,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAAkB,CAAIB,CAAX,G AAKC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,8D;MAAA,kD;QAWiB, Q;QAFb,IA59SO,qBAAQ,CA49Sf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,UAAK,CAAL,CAAT,C;QACF,+B;QA Ab,aAAU,CAAV,iB;UACI,QAAQ,SAAS,UAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAAkB,C AAIB,CAAX,GAAKc,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;sGAoBA,yB;MAAA,oC;MAA A,8D;MAAA,kD;QAWiB,Q;QAFb,IAx+SO,qBAAQ,CAw+Sf,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CA AL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,EAAT,C;UACR,IAAI,UAA W,SAAQ,QAAR,EAAkB,CAAIB,CAAX,GAAKc,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;IA oBA,8B;MASiB,Q;MAFb,IA1jTO,qBAAQ,CA0jTf,C;QAAe,OAAO,I;MACtB,UAAU,UAAK,CAAL,C;MACG,+ B;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,MkB1/hBG,MAAO,KIB0/hBE,GkB1/hBF,EIB0 /hBO,CkB1/hBP,C;;MIB4/hBd,OAAO,G;K;IAGX,gC;MASiB,Q;MAFb,IA1kTO,qBAAQ,CA0kTf,C;QAAe,OAA O,I;MACtB,UAAU,UAAK,CAAL,C;MACG,+B;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR, MkB1/hBG,MAAO,KIBqhiBE,GkB1/hBF,EIBqhiBO,CkB1/hBP,C;;MIBuhiBd,OAAO,G;K;IAGX,gC;MAOiB,Q;M AFb,IAxlTO,qBAAQ,CAwlTf,C;QAAe,OAAO,I;MACtB,UAAU,UAAK,CAAL,C;MACG,+B;MAAb,aAAU,CAA V,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,IAAI,sBAAM,CAAN,KAAJ,C;UAAa,MAAM,C;;MAEvB,OAAO,G; K;IAGX,gC;MAOiB,Q;MAFb,IA9lTO,qBAAQ,CA8lTf,C;QAAe,OAAO,I;MACtB,UAAU,UAAK,CAAL,C;MAC G,iC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,IAAI,MAAM,CAAV,C;UAAa,MAAM,C;;M AEvB,OAAO,G;K;IAGX,gC;MAOiB,Q;MAFb,IApmTO,qBAAQ,CAomTf,C;QAAe,OAAO,I;MACtB,UAAU,UA AK,CAAL,C;MACG,iC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,IAAI,MAAM,CAAV,C; UAAa,MAAM,C;;MAEvB,OAAO,G;K;IAGX,gC;MAOiB,Q;MAFb,IA1mTO,qBAAQ,CA0mTf,C;QAAe,OAAO,I ;MACtB,UAAU,UAAK,CAAL,C;MACG,iC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,IAAI ,MAAM,CAAV,C;UAAa,MAAM,C;;MAEvB,OAAO,G;K;IAGX,gC;MAOiB,Q;MAFb,IAhnTO,qBAAQ,CAgnTf, C;QAAe,OAAO,I;MACtB,UAAU,UAAK,CAAL,C;MACG,iC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CA AL,C;QACR,IAAI,oBAAM,CAAN,KAAJ,C;UAAa,MAAM,C;;MAEvB,OAAO,G;K;IAGX,gC;MASiB,Q;MAFb,IA xnTO,qBAAQ,CAwnTf,C;QAAe,OAAO,I;MACtB,UAAU,UAAK,CAAL,C;MACG,iC;MAAb,aAAU,CAAV,iB; QACI,QAAQ,UAAK,CAAL,C;QACR,MkB3miBG,MAAO,KIB2miBE,GkB3miBF,EIB2miBO,CkB3miBP,C;;MI B6miBd,OAAO,G;K;IAGX,gC;MASiB,Q;MAFb,IAhoTO,qBAAQ,CAgoTf,C;QAAe,OAAO,I;MACtB,UAAU,UA AK,CAAL,C;MACG,iC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,MkBhniBG,MAAO,KIBg niBE,GkBhniBF,EIBgniBO,CkBhniBP,C;;MIBkniBd,OAAO,G;K;IAGX,gC;MAOiB,Q;MAFb,IA9nTO,qBAAQ,C A8nTf,C;QAAe,OAAO,I;MACtB,UAAU,UAAK,CAAL,C;MACG,iC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UA AK,CAAL,C;QACR,IAAI,MAAM,CAAV,C;UAAa,MAAM,C;;MAEvB,OAAO,G;K;IAGX,wC;MAGI,OAAO,yB AAc,UAAAd,C;K;IAGX,0C;MAGI,OAAO,2BAAc,UAAAd,C;K;IAGX,0C;MAGI,OAAO,2BAAc,UAAAd,C;K;IAGX, 0C;MAGI,OAAO,2BAAc,UAAAd,C;K;IAGX,0C;MAGI,OAAO,2BAAc,UAAAd,C;K;IAGX,0C;MAGI,OAAO,2BA Ac,UAAAd,C;K;IAGX,0C;MAGI,OAAO,2BAAc,UAAAd,C;K;IAGX,0C;MAGI,OAAO,2BAAc,UAAAd,C;K;IAGX,0 C;MAGI,OAAO,2BAAc,UAAAd,C;K;IAGX,8C;MAOiB,Q;MAFb,IAlwTO,qBAAQ,CAkwTf,C;QAAe,OAAO,I;M ACtB,UAAU,UAAK,CAAL,C;MACG,+B;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,IAAI,U AAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IAGX,gD; MAOiB,Q;MAFb,IAxwTO,qBAAQ,CAwwTf,C;QAAe,OAAO,I;MACtB,UAAU,UAAK,CAAL,C;MACG,iC;MA Ab,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,IAAI,UAAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GA A6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IAGX,gD;MAOiB,Q;MAFb,IA9wTO,qBAAQ,CA8wTf, C;QAAe,OAAO,I;MACtB,UAAU,UAAK,CAAL,C;MACG,iC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CA AL,C;QACR,IAAI,UAAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,O AAO,G;K;IAGX,gD;MAOiB,Q;MAFb,IApxTO,qBAAQ,CAoxTf,C;QAAe,OAAO,I;MACtB,UAAU,UAAK,CAA L,C;MACG,iC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,IAAI,UAAW,SAAQ,GAAR,EAAa ,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IAGX,gD;MAOiB,Q;MAFb,IA1xTO, qBAAQ,CA0xTf,C;QAAe,OAAO,I;MACtB,UAAU,UAAK,CAAL,C;MACG,iC;MAAb,aAAU,CAAV,iB;QACI,Q AAQ,UAAK,CAAL,C;QACR,IAAI,UAAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAA

M,C;;MAE9C,OAAO,G;K;IAGX,gD;MAOiB,Q;MAFb,IAhyTO,qBAAQ,CAgyTf,C;QAAe,OAAO,I;MAcTb,UA  
AU,UAAK,CAAL,C;MACG,iC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,IAAI,UAAW,SA  
AQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IAGX,gD;MAOiB,  
Q;MAFb,IAtyTO,qBAAQ,CAsyTf,C;QAAe,OAAO,I;MAcTb,UAAU,UAAK,CAAL,C;MACG,iC;MAAb,aAAU,C  
AAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,IAAI,UAAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC  
,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IAGX,gD;MAOiB,Q;MAFb,IA5yTO,qBAAQ,CA4yTf,C;QAAe,OAA  
O,I;MAcTb,UAAU,UAAK,CAAL,C;MACG,iC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,I  
AAI,UAAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IA  
GX,gD;MAOiB,Q;MAFb,IAIzTO,qBAAQ,CAkzTf,C;QAAe,OAAO,I;MAcTb,UAAU,UAAK,CAAL,C;MACG,iC  
;MAAb,aAAU,CAAV,iB;QACI,QAAQ,UAAK,CAAL,C;QACR,IAAI,UAAW,SAAQ,gBAAR,EAAa,cAAb,CAA  
X,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IAGX,yB;MAMI,OAj4TO,qBAAQ,C;K;IAo4TnB,  
2B;MAMI,OA14TO,qBAAQ,C;K;IAq4TnB,2B;MAMI,OAn4TO,qBAAQ,C;K;IAS4TnB,2B;MAMI,OAp4TO,qBA  
AQ,C;K;IAu4TnB,2B;MAMI,OAr4TO,qBAAQ,C;K;IAw4TnB,2B;MAMI,OAt4TO,qBAAQ,C;K;IAY4TnB,2B;M  
AMI,OA44TO,qBAAQ,C;K;IA04TnB,2B;MAMI,OA44TO,qBAAQ,C;K;IA24TnB,2B;MAMI,OA44TO,qBAAQ,C  
;K;gFA44TnB,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OA  
AV,CAAJ,C;UAAwB,OAAO,K;;MACrD,OAAO,I;K;gFAGX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;QA  
AgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,K;;MACrD,OAAO,I;K;iFAGX,gC;  
MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAA  
wB,OAAO,K;;MACrD,OAAO,I;K;iFAGX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAgB,cAAA,SAAhB,  
M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,K;;MACrD,OAAO,I;K;iFAGX,gC;MAMoB,Q;MAAhB  
,wBAAGB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,K;;MAC  
rD,OAAO,I;K;iFAGX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,U  
AAU,OAAV,CAAJ,C;UAAwB,OAAO,K;;MACrD,OAAO,I;K;iFAGX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB  
,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,K;;MACrD,OAAO,I;K;iFA  
GX,gC;MAMoB,Q;MAAhB,wBAAGB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAAsB,IAAI,UAAU,OAAV,CAAJ  
,C;UAAwB,OAAO,K;;MACrD,OAAO,I;K;iFAGX,yB;MAAA,oC;MAAA,gC;MAAA,uC;QAMoB,Q;QAAhB,wB  
AAGB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UAAhB,IAAI,UAAU,oBAAV,CAAJ,C;YAAwB,OAAO,K  
;;QACrD,OAAO,I;O;KAPX,C;kFAUA,6B;MAMmC,Q;MAAhB,iD;QAAgB,cAAhB,e;QAAsB,OAAO,OAAP,C;;  
MAArC,gB;K;oFAGJ,6B;MAMmC,Q;MAAhB,iD;QAAgB,cAAhB,e;QAAsB,OAAO,OAAP,C;;MAArC,gB;K;oF  
AGJ,6B;MAMmC,Q;MAAhB,iD;QAAgB,cAAhB,e;QAAsB,OAAO,OAAP,C;;MAArC,gB;K;oFAGJ,6B;MAMmC  
,Q;MAAhB,iD;QAAgB,cAAhB,e;QAAsB,OAAO,OAAP,C;;MAArC,gB;K;oFAGJ,6B;MAMmC,Q;MAAhB,iD;Q  
AAGB,cAAhB,e;QAAsB,OAAO,OAAP,C;;MAArC,gB;K;oFAGJ,6B;MAMmC,Q;MAAhB,iD;QAAgB,cAAhB,e;  
QAAsB,OAAO,OAAP,C;;MAArC,gB;K;oFAGJ,6B;MAMmC,Q;MAAhB,iD;QAAgB,cAAhB,e;QAAsB,OAAO,O  
AAP,C;;MAArC,gB;K;oFAGJ,6B;MAMmC,Q;MAAhB,iD;QAAgB,cAAhB,e;QAAsB,OAAO,OAAP,C;;MAArC,  
gB;K;oFAGJ,yB;MAAA,oC;MAAA,gC;MAAA,oC;QAMmC,Q;QAAhB,iD;UAAgB,cAAhB,0B;UAAhB,OAAO,o  
BAAP,C;;QAArC,gB;O;KANJ,C;gGASA,6B;MARjJiB,gB;MADb,YAAY,C;MACZ,iD;QAAa,WAAb,e;QAAMb,  
QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;MA8jJnB,gB;K;kGAGJ,6B;MAvjJiB,gB;MADb,YAAY,C;MAC  
Z,iD;QAAa,WAAb,e;QAAMb,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;MAGkJnB,gB;K;kGAGJ,6B;MAz  
jJiB,gB;MADb,YAAY,C;MACZ,iD;QAAa,WAAb,e;QAAMb,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;M  
AkkJnB,gB;K;kGAGJ,6B;MA3jJiB,gB;MADb,YAAY,C;MACZ,iD;QAAa,WAAb,e;QAAMb,QAAO,cAAP,EAA  
O,sBAAP,WAAgB,IAAhB,C;;MAokJnB,gB;K;kGAGJ,6B;MA7jJiB,gB;MADb,YAAY,C;MACZ,iD;QAAa,WAA  
b,e;QAAMb,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;MAskJnB,gB;K;kGAGJ,6B;MAjJiB,gB;MADb,Y  
AAY,C;MACZ,iD;QAAa,WAAb,e;QAAMb,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;MAwkJnB,gB;K;k  
GAGJ,6B;MAjkJiB,gB;MADb,YAAY,C;MACZ,iD;QAAa,WAAb,e;QAAMb,QAAO,cAAP,EAAO,sBAAP,WAA  
gB,IAAhB,C;;MA0kJnB,gB;K;kGAGJ,6B;MAnkJiB,gB;MADb,YAAY,C;MACZ,iD;QAAa,WAAb,e;QAAMb,QA  
AO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;MA4kJnB,gB;K;kGAGJ,yB;MAAA,6B;MAAA,sC;MA5kJA,oC;M  
AAA,gC;MA4kJA,2BASiB,yB;QArLjB,oC;QAAA,gC;eAqLjB,0B;UAAA,4B;YAAE,aAAe,c;YA9kJjB,gB;YADb  
,YAAY,C;YACZ,iD;cAAa,WAAb,0B;cAAmB,QAAO,cAAP,EAAO,sBAAP,WAAgB,iBAAhB,C;;YA8kJmB,W;

W;S;OAAzB,C;MATjB,oC;QArkJiB,gB;QADb,YAAY,C;QACZ,iD;UAAa,WAAb,0B;UAAmB,QAAO,cAAP,EA  
AO,sBAAP,WAAgB,iBAAhB,C;;QA8kJnB,gB;O;KATJ,C;kFAYA,yB;MAAA,4F;MAAA,8D;MAAA,uC;QAgBq  
B,Q;QAHjB,IAhvUO,qBAAQ,CAGvUf,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAqB,UAAK,CAAL,C  
;QACJ,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,UAAK,KAAL,CAAvB,C;;QAEIB,OA  
AO,W;O;KAnBX,C;oFAsBA,yB;MAAA,4F;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IA9vUO,qBAAQ,CA8vU  
f,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;  
UACI,cAAc,UAAU,WAAV,EAAuB,UAAK,KAAL,CAAvB,C;;QAEIB,OAAO,W;O;KAnBX,C;oFAsBA,yB;MA  
AA,4F;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IA5wUO,qBAAQ,CA4wUf,C;UACI,MAAM,mCAA8B,+BAA  
9B,C;QACV,kBAAkB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAu  
B,UAAK,KAAL,CAAvB,C;;QAEIB,OAAO,W;O;KAnBX,C;oFAsBA,yB;MAAA,4F;MAAA,8D;MAAA,uC;QAg  
BqB,Q;QAHjB,IA1xUO,qBAAQ,CA0xUf,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,UAAK,CAA  
L,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,UAAK,KAAL,CAAvB,C;;QAEIB,  
OAAO,W;O;KAnBX,C;oFAsBA,yB;MAAA,4F;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IAxyUO,qBAAQ,CA  
wyUf,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd  
yB;UACI,cAAc,UAAU,WAAV,EAAuB,UAAK,KAAL,CAAvB,C;;QAEIB,OAAO,W;O;KAnBX,C;oFAsBA,yB;  
MAAA,4F;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IAtzUO,qBAAQ,CAszUf,C;UACI,MAAM,mCAA8B,+BA  
A9B,C;QACV,kBAAkB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAA  
uB,UAAK,KAAL,CAAvB,C;;QAEIB,OAAO,W;O;KAnBX,C;oFAsBA,yB;MAAA,4F;MAAA,8D;MAAA,uC;QA  
gBqB,Q;QAHjB,IAp0UO,qBAAQ,CAo0Uf,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,UAAK,CAA  
L,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,UAAK,KAAL,CAAvB,C;;QAEIB,  
OAAO,W;O;KAnBX,C;oFAsBA,yB;MAAA,4F;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IA11UO,qBAAQ,CAk  
1Uf,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,  
yB;UACI,cAAc,UAAU,WAAV,EAAuB,UAAK,KAAL,CAAvB,C;;QAEIB,OAAO,W;O;KAnBX,C;oFAsBA,yB;  
MAAA,4F;MAAA,8D;MAAA,oC;MAAA,gC;MAAA,uC;QAgBqB,Q;QAHjB,IAh2UO,qBAAQ,CAG2Uf,C;UACI,  
MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cA  
Ac,oBAAU,wBAAV,EAAuB,sBAAK,KAAL,EAAvB,E;;QAEIB,OAAO,W;O;KAnBX,C;gGAsBA,yB;MAAA,4F;  
MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IA7UO,qBAAQ,CAs7Uf,C;UACI,MAAM,mCAA8B,+BAA9B,C;QA  
CV,kBAAqB,UAAK,CAAL,C;QACJ,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAHV,EAAiB,WAAjB,E  
AA8B,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,C;kGAsBA,yB;MAAA,4F;MAAA,8D;MAAA,uC;  
QAgBqB,Q;QAHjB,IAp8UO,qBAAQ,CAo8Uf,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,UAAK,C  
AAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAHV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,  
CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,C;kGAsBA,yB;MAAA,4F;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,I  
A19UO,qBAAQ,CAk9Uf,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,UAAK,CAAL,C;QACD,+B;Q  
AAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAHV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;;QAEIB,O  
AAO,W;O;KAnBX,C;kGAsBA,yB;MAAA,4F;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IAh+UO,qBAAQ,CAG  
+Uf,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,  
yB;UACI,cAAc,UAAU,KAHV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,  
C;kGAsBA,yB;MAAA,4F;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IA9+UO,qBAAQ,CA8+Uf,C;UACI,MAA  
M,mCAA8B,+BAA9B,C;QACV,kBAAkB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UA  
AU,KAHV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,C;kGAsBA,yB;MA  
AA,4F;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IA5/UO,qBAAQ,CA4/Uf,C;UACI,MAAM,mCAA8B,+BAA9  
B,C;QACV,kBAAkB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAHV,EAAiB,  
WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,C;kGAsBA,yB;MAAA,4F;MAAA,8D;  
MAAA,uC;QAgBqB,Q;QAHjB,IA1gVO,qBAAQ,CA0gVf,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAA  
kB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAHV,EAAiB,WAAjB,EAA8B,U  
AAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,C;kGAsBA,yB;MAAA,4F;MAAA,8D;MAAA,uC;QAgBqB  
,Q;QAHjB,IAxhVO,qBAAQ,CAwhVf,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,UAAK,CAAL,C;  
QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAHV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B

,C;;QAEIB,OAAO,W;O;KAnBX,C;kGAsBA,yB;MAAA,4F;MAAA,8D;MAAA,oC;MAAA,gC;MAAA,uC;QAgBqB,Q;QAHjB,IAtiVO,qBAAQ,CAsiVf,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,oBAAU,KAAV,EAAiB,wBAAjB,EAA8B,sBAAK,KAAL,EAA9B,E;;QAEIB,OAAO,W;O;KAnBX,C;4GAsBA,yB;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IA5nVO,qBAAQ,CA4nVf,C;UACI,OAAO,I;QACX,kBAAqB,UAAK,CAAL,C;QACJ,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,C;8GAsBA,yB;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IA1oVO,qBAAQ,CA0oVf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,C;8GAsBA,yB;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IAxpVO,qBAAQ,CAwpVf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,C;8GAsBA,yB;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IA1oVO,qBAAQ,CAsqVf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,C;8GAsBA,yB;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IAIsVO,qBAAQ,CAksVf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,C;8GAsBA,yB;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IAhtVO,qBAAQ,CAgtVf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,C;8GAsBA,yB;MAAA,8D;MAAA,uC;QAgBqB,Q;QAHjB,IA9tVO,qBAAQ,CA8tVf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAnBX,C;8GAsBA,yB;MAAA,8D;MAAA,oC;MAAA,gC;MAAA,uC;QAgBqB,Q;QAHjB,IA5uVO,qBAAQ,CA4uVf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,oBAAU,KAAV,EAAiB,wBAAjB,EAA8B,sBAAK,KAAL,EAA9B,E;;QAEIB,OAAO,W;O;KAnBX,C;8FAsBA,yB;MAAA,8D;MAAA,uC;QAIbqB,Q;QAHjB,IAN0VO,qBAAQ,CAM0Vf,C;UACI,OAAO,I;QACX,kBAAqB,UAAK,CAAL,C;QACJ,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KApBX,C;gGAuBA,yB;MAAA,8D;MAAA,uC;QAIbqB,Q;QAHjB,IAL1VO,qBAAQ,CAk1Vf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KApBX,C;gGAuBA,yB;MAAA,8D;MAAA,uC;QAIbqB,Q;QAHjB,IAj2VO,qBAAQ,CAi2Vf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KApBX,C;gGAuBA,yB;MAAA,8D;MAAA,uC;QAIbqB,Q;QAHjB,IAh3VO,qBAAQ,CAg3Vf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KApBX,C;gGAuBA,yB;MAAA,8D;MAAA,uC;QAIbqB,Q;QAHjB,IA75VO,qBAAQ,CA65Vf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KApBX,C;gGAuBA,yB;MAAA,8D;MAAA,uC;QAIbqB,Q;QAHjB,IA56VO,qBAAQ,CA46Vf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,UAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KApBX,C;gGAuBA,yB;MAAA,8D;MAAA,oC;MAAA,gC;MAAA,uC;QAIbqB,Q;QAHjB,IA37VO,qBAAQ,CA27Vf,C;UACI,OAAO,I;QACX,kBAakB,UAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,oBAAU,wBAAV,EAAuB,sBAAK,KAAL,EAA9B,E;;QAEIB,OAAO,W;O;KApBX,C;4FAuBA,yB;M

AAA,8D;MAAA,4F;MAAA,uC;QAe6B,UAE0,M;QAJhC,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAA  
M,mCAA8B,+BAA9B,C;QACrB,kBAAqB,UAAI,YAAJ,EAAl,oBAAJ,O;QACrB,OAAO,SAAS,CAAhB,C;UACI,  
cAAc,UAAU,UAAI,cAAJ,EAAl,sBAAJ,SAAV,EAawB,WAAxB,C;;QAEIB,OAAO,W;O;KAnBX,C;8FAsBA,yB  
;MAAA,8D;MAAA,4F;MAAA,uC;QAe0B,UAEU,M;QAJhC,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,M  
AAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,UAAI,YAAJ,EAAl,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;U  
ACI,cAAc,UAAU,UAAI,cAAJ,EAAl,sBAAJ,SAAV,EAawB,WAAxB,C;;QAEIB,OAAO,W;O;KAnBX,C;8FAsB  
A,yB;MAAA,8D;MAAA,4F;MAAA,uC;QAe0B,UAEU,M;QAJhC,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAA  
e,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,UAAI,YAAJ,EAAl,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C  
;UACI,cAAc,UAAU,UAAI,cAAJ,EAAl,sBAAJ,SAAV,EAawB,WAAxB,C;;QAEIB,OAAO,W;O;KAnBX,C;8FAs  
BA,yB;MAAA,8D;MAAA,4F;MAAA,uC;QAe0B,UAEU,M;QAJhC,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UA  
Ae,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,UAAI,YAAJ,EAAl,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB  
,C;UACI,cAAc,UAAU,UAAI,cAAJ,EAAl,sBAAJ,SAAV,EAawB,WAAxB,C;;QAEIB,OAAO,W;O;KAnBX,C;8F  
AsBA,yB;MAAA,8D;MAAA,4F;MAAA,uC;QAe0B,UAEU,M;QAJhC,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;  
UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,UAAI,YAAJ,EAAl,oBAAJ,O;QACIB,OAAO,SAAS,CA  
AhB,C;UACI,cAAc,UAAU,UAAI,cAAJ,EAAl,sBAAJ,SAAV,EAawB,WAAxB,C;;QAEIB,OAAO,W;O;KAn  
BX,C;8FAsBA,yB;MAAA,8D;MAAA,4F;MAAA,uC;QAe0B,UAEU,M;QAJhC,YAAY,wB;QACZ,IAAI,QAAQ,CAA  
Z,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,UAAI,YAAJ,EAAl,oBAAJ,O;QACIB,OAAO,SAAS,  
CAAhB,C;UACI,cAAc,UAAU,UAAI,cAAJ,EAAl,sBAAJ,SAAV,EAawB,WAAxB,C;;QAEIB,OAAO,W;O;KAn  
BX,C;8FAsBA,yB;MAAA,8D;MAAA,4F;MAAA,uC;QAe0B,UAEU,M;QAJhC,YAAY,wB;QACZ,IAAI,QAAQ,C  
AAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,UAAI,YAAJ,EAAl,oBAAJ,O;QACIB,OAAO,SA  
AS,CAAhB,C;UACI,cAAc,UAAU,UAAI,cAAJ,EAAl,sBAAJ,SAAV,EAawB,WAAxB,C;;QAEIB,OAAO,W;O;K  
AnBX,C;8FAsBA,yB;MAAA,8D;MAAA,4F;MAAA,uC;QAe0B,UAEU,M;QAJhC,YAAY,wB;QACZ,IAAI,QAA  
Q,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,UAAI,YAAJ,EAAl,oBAAJ,O;QACIB,OAAO  
,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,cAAJ,EAAl,sBAAJ,SAAV,EAawB,WAAxB,C;;QAEIB,OAAO,W;  
O;KAnBX,C;8FAsBA,yB;MAAA,8D;MAAA,4F;MAAA,oC;MAAA,gC;MAAA,uC;QAe0B,UAEU,M;QAJhC,YA  
AY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,UAAI,YAAJ,EAAl  
,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,oBAAU,sBAAl,cAAJ,EAAl,sBAAJ,UAAV,EAawB,wB  
AAxB,E;;QAEIB,OAAO,W;O;KAnBX,C;0GAsBA,yB;MAAA,8D;MAAA,4F;MAAA,uC;QAe6B,Q;QAFzB,YAA  
Y,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAAqB,UAAI,YAAJ,EAAl,o  
BAAJ,O;QACrB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAl,iB,UAAI,KAAJ,CAAjB,EAA6B,WA  
A7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;4GAuBA,yB;MAAA,8D;MAAA,4F;MAAA,uC;QAe0B,Q;QAFt  
B,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,UAAI,YAAJ,  
EAAl,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAl,iB,UAAI,KAAJ,CAAjB,EAA  
6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;4GAuBA,yB;MAAA,8D;MAAA,4F;MAAA,uC;QAe0B,  
Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,UAA  
I,YAAJ,EAAl,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAl,iB,UAAI,KAAJ,CAAj  
B,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;4GAuBA,yB;MAAA,8D;MAAA,4F;MAAA,uC;Q  
Ae0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB  
,UAAI,YAAJ,EAAl,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAl,iB,UAAI,KAAJ,  
CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;4GAuBA,yB;MAAA,8D;MAAA,4F;MAAA  
,uC;QAe0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kB  
AAkB,UAAI,YAAJ,EAAl,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAl,iB,UAAI,  
KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;4GAuBA,yB;MAAA,8D;MAAA,4F;  
MAAA,uC;QAe0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QA  
CrB,kBAakB,UAAI,YAAJ,EAAl,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAl,iB  
,UAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;4GAuBA,yB;MAAA,8D;MA  
AA,4F;MAAA,uC;QAe0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9  
B,C;QACrB,kBAakB,UAAI,YAAJ,EAAl,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV

,EAAiB,UAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;4GAuBA,yB;MAAA,8D;MAAA,4F;MAAA,uC;QAE0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,UAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;4GAuBA,yB;MAAA,8D;MAAA,4F;MAAA,oC;MAAA,gC;MAAA,uC;QAE0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,oBAAU,KAAV,EAAiB,sBAAI,KAAJ,EAAjB,EAA6B,wBAA7B,E;UACd,qB;;QAEJ,OA AO,W;O;KApBX,C;sHAuBA,yB;MAAA,8D;MAAA,uC;QAE6B,Q;QAFzB,YAAY,wB;QACZ,IAAI,QAAQ,CAA Z,C;UAAe,OAAO,I;QACtB,kBAAqB,UAAI,YAAJ,EAAI,oBAAJ,O;QACrB,OAAO,SAAS,CAAhB,C;UACI,cAA c,UAAU,KAAV,EAAiB,UAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;wHA uBA,yB;MAAA,8D;MAAA,uC;QAE0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QAC tB,kBAakB,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB, UAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;wHAuBA,yB;MAAA,8D;MA AA,uC;QAE0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,UAAI,YAA J,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,UAAI,KAAJ,CAAjB,EA A6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;wHAuBA,yB;MAAA,8D;MAAA,uC;QAE0B,Q;QAFtB ,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,UAAI,YAAJ,EAAI,oBAAJ,O;QACI B,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,UAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,q B;;QAEJ,OAAO,W;O;KApBX,C;wHAuBA,yB;MAAA,8D;MAAA,uC;QAE0B,Q;QAFtB,YAAY,wB;QACZ,IAAI, QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB, C;UACI,cAAc,UAAU,KAAV,EAAiB,UAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,q B;;QAEJ,OAAO,W;O;K ApBX,C;wHAuBA,yB;MAAA,8D;MAAA,uC;QAE0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe, OAAO,I;QACtB,kBAakB,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB, C;UACI,cAAc,UAAU,KAAV,EAAiB,UAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,q B;;QAEJ,OAAO,W;O;K ApBX,C;wHAuBA,yB;MAAA,8D;MAAA,uC;QAE0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe, OAAO,I;QACtB,kBAakB,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,K AAV,EAAiB,UAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;wHAuBA,yB;M AAA,8D;MAAA,uC;QAE0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAak B,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,UAAI,KAA J,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;wHAuBA,yB;MAAA,8D;MAAA,uC;QAE 0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,UAAI,YAAJ,EAAI,oB AAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,UAAI,KAAJ,CAAjB,EAA6B,WAA 7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;wHAuBA,yB;MAAA,8D;MAAA,uC;QAE 0B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,UAAI,YAAJ,EAAI,oB AAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,UAAI,KAAJ,CAAjB,EAA6B,WAA 7B,C;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;wHAuBA,yB;MAAA,8D;MAAA,oC;MAAA,gC;MAAA,uC;QAE0 B,Q;QAFtB,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,UAAI,YAAJ,EAAI,oBA AJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,oBAAU,KAAV,EAAiB,sBAAI,KAAJ,EAAjB,EAA6B,wBAA 7B,E;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;wGAuBA,yB;MAAA,8D;MAAA,uC;QAgB0B,UAEU,M;QAJhC, YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAAqB,UAAI,YAAJ,EAAI,oBAAJ,O;QACrB ,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,cAAJ,EAAI,sBAAJ,SAAV,EAAwB,WAAxB,C;;QAEIB,OA AO,W;O;KApBX,C;0GAuBA,yB;MAAA,8D;MAAA,uC;QAgB0B,UAEU,M;QAJhC,YAAY,wB;QACZ,IAAI,QA AQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;U ACI,cAAc,UAAU,UAAI,cAAJ,EAAI,sBAAJ,SAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KApBX,C;0GAuB A,yB;MAAA,8D;MAAA,uC;QAgB0B,UAEU,M;QAJhC,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO, I;QACtB,kBAakB,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,cA AJ,EAAI,sBAAJ,SAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KApBX,C;0GAuBA,yB;MAAA,8D;MAAA,uC ;QAgB0B,UAEU,M;QAJhC,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,UAAI,Y AAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,cAAJ,EAAI,sBAAJ,SAAV,EA AwB,WAAxB,C;;QAEIB,OAAO,W;O;KApBX,C;0GAuBA,yB;MAAA,8D;MAAA,uC;QAgB0B,UAEU,M;QAJh C,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,UAAI,YAAJ,EAAI,oBAAJ,O;QAC IB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,cAAJ,EAAI,sBAAJ,SAAV,EAAwB,WAAxB,C;;QAEIB,O AAO,W;O;KApBX,C;0GAuBA,yB;MAAA,8D;MAAA,uC;QAgB0B,UAEU,M;QAJhC,YAAY,wB;QACZ,IAAI,Q AAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;

UACI,cAAc,UAAU,UAAI,cAAJ,EAAI,sBAAJ,SAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KApBX,C;0GAuBA,yB;MAAA,8D;MAAA,uC;QAgB0B,UAEU,M;QAJhC,YAA Y,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAAkB,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,cAAJ,EAAI,sBAAJ,SAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KApBX,C;0GAuBA,yB;MAAA,8D;MAAA,uC;QAgB0B,UAEU,M;QAJhC,YAA Y,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAAkB,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,UAAI,cAAJ,EAAI,sBAAJ,SAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KApBX,C;0GAuBA,yB;MAAA,8D;MAAA,oC;MAAA,gC;MAAA,uC;QAgB0B,UAEU,M;QAJhC,YAA Y,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAAkB,UAAI,YAAJ,EAAI,oBAAJ,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,oBAAU,sBAAI,cAAJ,EAAI,sBAAJ,UAAV,EAAwB,wBAAxB,E;;QAEIB,OAAO,W;O;KApBX,C;4FAuBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAgBoB,Q;QAHhB,IAp0XO,qBAAQ,CAo0Xf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBj9mBO,W;QjBk9mBP,kBAAkB,O;QACIB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KApBX,C;8FAuBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAIBoB,Q;QAHhB,IAp1XO,qBAAQ,CAo1Xf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBz+mBO,W;QjB0+mBP,kBAAkB,O;QACIB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KArBX,C;8FAwBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAIBoB,Q;QAHhB,IAp2XO,qBAAQ,CAo2Xf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBjgnBO,W;QjBkgnBP,kBAAkB,O;QACIB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KArBX,C;8FAwBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAIBoB,Q;QAHhB,IAp3XO,qBAAQ,CAo3Xf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBzhnBO,W;QjB0hnBP,kBAAkB,O;QACIB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KArBX,C;8FAwBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAIBoB,Q;QAHhB,IAp4XO,qBAAQ,CAo4Xf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBjgnBO,W;QjBkgnBP,kBAAkB,O;QACIB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KArBX,C;8FAwBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAIBoB,Q;QAHhB,IAp5XO,qBAAQ,CAo5Xf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBzknBO,W;QjB0knBP,kBAAkB,O;QACIB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KArBX,C;8FAwBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAIBoB,Q;QAHhB,IAp6XO,qBAAQ,CAo6Xf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBjmnBO,W;QjBkmnBP,kBAAkB,O;QACIB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KArBX,C;8FAwBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAIBoB,Q;QAHhB,IAp7XO,qBAAQ,CAo7Xf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBjpnBO,W;QjBkpnBP,kBAAkB,O;QACIB,wBAAgB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UACI,cAAc,UAAU,WAAV,EAAuB,oBAAvB,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KArBX,C;0GAwBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QACI,IA5hYO,qBAAQ,CA4hYf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBzqnBO,W;QjB0qnBP,kBAAkB,O;QACIB,wD;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KArBX,C;4GAwBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAEI,IA7iYO,qBAAQ,CA6iYf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBlsnBO,W;QjBmsnBP,kBAAkB,O;QACIB,wD;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9

B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAtBX,C;4GAyBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAeI,IA9jYO,qBAAQ,CA8jYf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiB3tnBO,W;QjB4tnBP,kBAAkB,O;QACIB,wD;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAtBX,C;4GAyBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAeI,IA/kYO,qBAAQ,CA+kYf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBpvnBO,W;QjBqvnBP,kBAAkB,O;QACIB,wD;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAtBX,C;4GAyBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAeI,IAhmYO,qBAAQ,CAGmYf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiB7wnBO,W;QjB8wnBP,kBAAkB,O;QACIB,wD;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAtBX,C;4GAyBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAeI,IAjnYO,qBAAQ,CAinYf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBtynBO,W;QjBuynBP,kBAAkB,O;QACIB,wD;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAtBX,C;4GAyBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAeI,IAloYO,qBAAQ,CAkoYf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiB/znBO,W;QjBg0nBP,kBAAkB,O;QACIB,wD;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAtBX,C;4GAyBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAeI,IANpYO,qBAAQ,CAnpYf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBx1nBO,W;QjBy1nBP,kBAAkB,O;QACIB,wD;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAtBX,C;4GAyBA,yB;MAAA,gD;MAAA,gE;MAAA,oC;MAAA,gD;QAeI,IApqYO,qBAAQ,CAoqYf,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,aiBj3nBO,W;QjBk3nBP,kBAAkB,O;QACIB,wD;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,EAA9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAtBX,C;4GAyBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QACl,IA5vYO,qBAAQ,CA4vYf,C;UAAe,OAAO,W;QACtB,sBAaQb,UAAK,CAAL,CAArB,C;QACgC,kBAAnB,eAAa,gBAAb,C;QAA2B,sBAAl,aAAJ,C;QAAxC,aiB14nBO,W;QjB24nBP,iBAAc,CAAd,UAAAsB,gBAAtB,U;UACI,gBAAc,UAAU,aAAV,EAAuB,UAAK,KAAL,CAAvB,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KArBX,C;kGAwBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QAWI,IAzwYO,qBAAQ,CaywYf,C;UAAe,OAAO,W;QACtB,sBAaKB,UAAK,CAAL,CAAIb,C;QACmC,kBAAtB,eAAgB,gBAAhB,C;QAA8B,sBAAl,aAAJ,C;QAA3C,aiB/5nBO,W;QjBg6nBP,iBAAc,CAAd,UAAAsB,gBAAtB,U;UACI,gBAAc,UAAU,aAAV,EAAuB,UAAK,KAAL,CAAvB,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAIbX,C;kGAqBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QAWI,IAtxYO,qBAAQ,CAsxYf,C;UAAe,OAAO,W;QACtB,sBAaKB,UAAK,CAAL,CAAIb,C;QACoC,kBAAvB,eAAiB,gBAAjB,C;QAA+B,sBAAl,aAAJ,C;QAA5C,aiBp7nBO,W;QjBq7nBP,iBAAc,CAAd,UAAAsB,gBAAtB,U;UACI,gBAAc,UAAU,aAAV,EAAuB,UAAK,KAAL,CAAvB,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAIbX,C;kGAqBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QAWI,IAAnyYO,qBAAQ,CAnyYf,C;UAAe,OAAO,W;QACtB,sBAaKB,UAAK,CAAL,CAAIb,C;QACkC,kBAArB,eAAe,gBAAf,C;QAA6B,sBAAl,aAAJ,C;QAA1C,aiBz8nBO,W;QjB08nBP,iBAAc,CAAd,UAAAsB,gBAAtB,U;UACI,gBAAc,UAAU,aAAV,EAAuB,UAAK,KAAL,CAAvB,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAIbX,C;kGAqBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QAWI,IAhzYO,qBAAQ,CAGzYf,C;UAAe,OAAO,W;QACtB,sBAaKB,UAAK,CAAL,CAAIb,C;QACmC,kBAAtB,eAAgB,gBAAhB,C;QAA8B,sBAAl,aAAJ,C;QAA3C,aiB99nBO,W;QjB+9nBP,iBAAc,CAAd,UAAAsB,gBAAtB,U;UACI,gBAAc,UAAU,aAAV,EAAuB,UAAK,KAAL,CAAvB,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAIbX,C;kGAqBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QAWI,IA7zYO,qBAAQ,CA6zYf,C;UAAe,OAAO,W;QACtB,sBAaKB,UAAK,CAAL,CAAIb,C;QACoC,kBAAvB,eAAiB,gBAAjB,C;QAA+B,sBAAl,aAAJ,C;QAA5C,aiBn/nBO,W;QjBo/nBP,iBAAc,CAAd,UAAAsB,gBAAtB,U;UACI,gBAAc,UAAU,aAAV,EAAuB,UAAK,KAAL,CAAvB,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAIbX,C;kGAqBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QAWI,IA10YO,qBAAQ,CA00Yf,C;UAAe,OAAO,W;QACtB,sBAaKB,UAAK,CAAL,CAAIb,C;QACqC,kBAAxB,eAAkB,gBAAIb,C;QAAgC,sBAAl,aAAJ,C;QAA7C,aiBxgoBO,W;QjBygoBP,iBAAc,CAAd,UAAAsB,gBAAtB,U;UACI,gBAAc,UAAU,aAAV,EAAuB,UAA

K,KAAL,CAA vB,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAIBX,C;kGAqBA,yB;MAAA,qD;MAA A,gE;MAAA,uC;QAWI,IAv1YO,qBAAQ,CAu1Yf,C;UAAe,OAAO,W;QACtB,sBAakB,UAAK,CAAL,CAAIB,C ;QACsC,kBAAZB,eAAmB,gBAAnB,C;QAAiC,sBAAI,aAAJ,C;QAA9C,aiB7hoBO,W;QjB8hoBP,iBAAc,CAAd,U AAsB,gBAAtB,U;UACI,gBAAc,UAAU,aAAV,EAAuB,UAAK,KAAL,CAA vB,C;UACd,MAAO,WAAI,aAAJ,C;; QAEX,OAAO,M;O;KAIBX,C;kGAqBA,yB;MAAA,qD;MAAA,gE;MAAA,oC;MAAA,gC;MAAA,uC;QAWI,IAP 2YO,qBAAQ,CAo2Yf,C;UAAe,OAAO,W;QACtB,sBAakB,UAAK,CAAL,CAAIB,C;QACmC,kBAAtB,eAAgB,g BAAhB,C;QAA8B,sBAAI,0BAAJ,C;QAA3C,aiBljoBO,W;QjBmjoBP,iBAAc,CAAd,UAA sB,gBAAtB,U;UACI,g BAAC,oBAAU,0BAAV,EAAuB,sBAAK,KAAL,EAA vB,E;UACd,MAAO,WAAI,0BAAJ,C;;QAEX,OAAO,M;O; KAIBX,C;8GAqBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QACI,IA57YO,qBAAQ,CA47Yf,C;UAAe,OAAO,W;Q ACtB,sBAaqB,UAAK,CAAL,CAArB,C;QACgC,kBAAnB,eAAa,gBAAb,C;QAA2B,sBAAI,aAAJ,C;QAAxC,aiB 1koBO,W;QjB2koBP,iBAAc,CAAd,UAA sB,gBAAtB,U;UACI,gBAAc,UAAU,KA AV,EAAiB,aAAjB,EAA8B,UA AK,KAAL,CAA9B,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KArBX,C;gHAWBA,yB;MAAA,qD;M AAA,gE;MAAA,uC;QAYI,IA18YO,qBAAQ,CA08Yf,C;UAAe,OAAO,W;QACtB,sBAakB,UAAK,CAAL,CAA I B,C;QACmC,kBAAtB,eAAgB,gBAAhB,C;QAA8B,sBAAI,aAAJ,C;QAA3C,aiBhmoBO,W;QjBimoBP,iBAAc,CA Ad,UAA sB,gBAAtB,U;UACI,gBAAc,UAAU,KA AV,EAAiB,aAAjB,EAA8B,UAAK,KAAL,CAA9B,C;UACd,M AAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAnBX,C;gHAsBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QAYI,IAx9 YO,qBAAQ,CAw9Yf,C;UAAe,OAAO,W;QACtB,sBAakB,UAAK,CAAL,CAAIB,C;QACoC,kBAAvB,eAAiB,gB AAjB,C;QAA+B,sBAAI,aAAJ,C;QAA5C,aiBtnoBO,W;QjBunoBP,iBAAc,CAAd,UAA sB,gBAAtB,U;UACI,gBA Ac,UAAU,KA AV,EAAiB,aAAjB,EAA8B,UAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAA O,M;O;KAnBX,C;gHAsBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QAYI,IA+YO,qBAAQ,CAs+Yf,C;UAAe,OA AO,W;QACtB,sBAakB,UAAK,CAAL,CAAIB,C;QACkC,kBAArB,eAAe,gBAAf,C;QAA6B,sBAAI,aAAJ,C;QA A1C,aiB5ooBO,W;QjB6ooBP,iBAAc,CAAd,UAA sB,gBAAtB,U;UACI,gBAAc,UAAU,KA AV,EAAiB,aAAjB,EAA8B,UAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAnBX,C;gHAsBA,yB;MAA A,qD;MAAA,gE;MAAA,uC;QAYI,IAP/YO,qBAAQ,CAo/Yf,C;UAAe,OAAO,W;QACtB,sBAakB,UAAK,CAAL, CAAIB,C;QACmC,kBAAtB,eAAgB,gBAAhB,C;QAA8B,sBAAI,aAAJ,C;QAA3C,aiBlqoBO,W;QjBmqoBP,iBAA c,CAAd,UAA sB,gBAAtB,U;UACI,gBAAc,UAAU,KA AV,EAAiB,aAAjB,EAA8B,UAAK,KAAL,CAA9B,C;UAC d,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAnBX,C;gHAsBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QAYI,I AlgZO,qBAAQ,CAkgZf,C;UAAe,OAAO,W;QACtB,sBAakB,UAAK,CAAL,CAAIB,C;QACoC,kBAAvB,eAAiB, gBAAjB,C;QAA+B,sBAAI,aAAJ,C;QAA5C,aiBxroBO,W;QjByroBP,iBAAc,CAAd,UAA sB,gBAAtB,U;UACI,gB AAc,UAAU,KA AV,EAAiB,aAAjB,EAA8B,UAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OA AO,M;O;KAnBX,C;gHAsBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QAYI,IAhhZO,qBAAQ,CAghZf,C;UAAe,O AAO,W;QACtB,sBAakB,UAAK,CAAL,CAAIB,C;QACqC,kBAAxB,eAAkB,gBAAlB,C;QAAgC,sBAAI,aAAJ,C ;QAA7C,aiB9soBO,W;QjB+soBP,iBAAc,CAAd,UAA sB,gBAAtB,U;UACI,gBAAc,UAAU,KA AV,EAAiB,aAAjB ,EAA8B,UAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAnBX,C;gHAsBA,yB;M AAA,qD;MAAA,gE;MAAA,uC;QAYI,IA9hZO,qBAAQ,CA8hZf,C;UAAe,OAAO,W;QACtB,sBAakB,UAAK,C AAL,CAAIB,C;QACsC,kBAAZB,eAAmB,gBAAnB,C;QAAiC,sBAAI,aAAJ,C;QAA9C,aiBpuoBO,W;QjBpuoBP,i BAAc,CAAd,UAA sB,gBAAtB,U;UACI,gBAAc,UAAU,KA AV,EAAiB,aAAjB,EAA8B,UAAK,KAAL,CAA9B,C; UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAnBX,C;gHAsBA,yB;MAAA,qD;MAAA,gE;MAAA,oC;M AAA,gC;MAAA,uC;QAYI,IA5iZO,qBAAQ,CA4iZf,C;UAAe,OAAO,W;QACtB,sBAakB,UAAK,CAAL,CAAIB, C;QACmC,kBAAtB,eAAgB,gBAAhB,C;QAA8B,sBAAI,0BAAJ,C;QAA3C,aiB1voBO,W;QjB2voBP,iBAAc,CA Ad,UAA sB,gBAAtB,U;UACI,gBAAc,oBAAU,KA AV,EAAiB,0BAAjB,EAA8B,sBAAK,KAAL,EAA9B,E;UACd, MAAO,WAAI,0BAAJ,C;;QAEX,OAAO,M;O;KAnBX,C;8EAsBA,yB;MA/zBA,gD;MAAA,gE;MA+zBA,gD;QAc W,sB;;UA7zBS,Q;UAHhB,IAP0XO,qBAAQ,CAoXf,C;YAAe,qBAAO,OAgoBH,OAhoBG,C;YAAP,uB;WACqB ,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBA+zBzB,OA/zByB,C;UAA5C,aiBj9mBO,W;UjBk9mBP,kB A8zBmB,O;UA7zBnB,iD;YAAgB,cAAhB,e;YACI,cA4zBwB,SA5zBV,CAAU,WAAV,EAAuB,OAAvB,C;YACd, MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;QAyzBP,yB;O;KADJ,C;gFAiBA,yB;MAzzBA,gD;MAAA,gE;MAyz BA,gD;QAeW,sB;;UA7zBS,Q;UAHhB,IAP1XO,qBAAQ,CAo1Xf,C;YAAe,qBAAO,OA0zBH,OA1zBG,C;YAAP, uB;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBAyzBzB,OAzzByB,C;UAA5C,aiBz+mBO,W;Uj

B0+mBP,kBAwzBmB,O;UAvzBnB,iD;YAAgB,cAAhB,e;YACl,cAszBwB,SAtzBV,CAAU,WAAV,EAAuB,OAAvB,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;;QAmzBP,yB;O;KafJ,C;gFAkBA,yB;MANzBA,gD;MAAA,gE;MAmzBA,gD;QAeW,sB;;UAjzBS,Q;UAHhB,IAP2XO,qBAAQ,CAo2Xf,C;YAAe,qBAAO,OAozBH,OApzBG,C;YAAP,uB;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBA mzBzB,OAnzByB,C;UAA5C,aiBjgnBO,W;UjBkgnBP,kBAkzBmB,O;UAjzBnB,iD;YAAgB,cAAhB,e;YACl,cAgzBwB,SAhzBV,CAAU,WAAV,EAAuB,OAAvB,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;;QA6yBP,yB;O;KafJ,C;gFAkBA,yB;MA7yBA,gD;MAAA,gE;MA6yBA,gD;QAeW,sB;;UA3yBS,Q;UAHhB,IAP3XO,qBAAQ,CAo3Xf,C;YAAe,qBAAO,OA8yBH,OA9yBG,C;YAAP,uB;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBA6yBzB,OA7yByB,C;UAA5C,aiBzhnBO,W;UjB0hnBP,kBA4yBmB,O;UA3yBnB,iD;YAAgB,cAAhB,e;YACl,cA0yBwB,SA1yBV,CAAU,WAAV,EAAuB,OAAvB,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;;QAuyBP,yB;O;KafJ,C;gFAkBA,yB;MAvyBA,gD;MAAA,gE;MAuyBA,gD;QAeW,sB;;UAryBS,Q;UAHhB,IAP4XO,qBAAQ,CAo4Xf,C;YAAe,qBAAO,OAwyBH,OAxyBG,C;YAAP,uB;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBAuyBzB,OAvyByB,C;UAA5C,aiBjgnBO,W;UjBkgnBP,kBAasyBmB,O;UAryBnB,iD;YAAgB,cAAhB,e;YACl,cAoyBwB,SApyBV,CAAU,WAAV,EAAuB,OAAvB,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;;QAiyBP,yB;O;KafJ,C;gFAkBA,yB;MAjyBA,gD;MAAA,gE;MAiyBA,gD;QAeW,sB;;UA/xBS,Q;UAHhB,IAP5XO,qBAAQ,CAo5Xf,C;YAAe,qBAAO,OAkyBH,OAlyBG,C;YAAP,uB;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBAiyBzB,OAjyByB,C;UAA5C,aiBzknBO,W;UjB0knBP,kBAgyBmB,O;UA/xBnB,iD;YAAgB,cAAhB,e;YACl,cA8xBwB,SA9xBV,CAAU,WAAV,EAAuB,OAAvB,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;;QA2xBP,yB;O;KafJ,C;gFAkBA,yB;MA3xBA,gD;MAAA,gE;MA2xBA,gD;QAeW,sB;;UAzxBS,Q;UAHhB,IAP6XO,qBAAQ,CAo6Xf,C;YAAe,qBAAO,OA4xBH,OA5xBG,C;YAAP,uB;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBA2xBzB,OA3xByB,C;UAA5C,aiBjmnBO,W;UjBkmnBP,kBA0xBmB,O;UAzxBnB,iD;YAAgB,cAAhB,e;YACl,cAwxBwB,SAxxBV,CAAU,WAAV,EAAuB,OAAvB,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;;QAqxBP,yB;O;KafJ,C;gFAkBA,yB;MARxBA,gD;MAAA,gE;MAqxBA,gD;QAeW,sB;;UANxBS,Q;UAHhB,IAP7XO,qBAAQ,CAo7Xf,C;YAAe,qBAAO,OA sxBH,OA txBG,C;YAAP,uB;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBAqxBzB,OA rxByB,C;UAA5C,aiBznnBO,W;UjB0nnBP,kBAoxBmB,O;UANxBnB,iD;YAAgB,cAAhB,e;YACl,cAkxBwB,SA1xBV,CAAU,WAAV,EAAuB,OAAvB,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;;QA+wBP,yB;O;KafJ,C;gFAkBA,yB;MA/wBA,gD;MAAA,gE;MAAA,oC;MAAA,gC;MA+wBA,gD;QAeW,sB;;UA7wBS,Q;UAHhB,IAP8XO,qBAAQ,CAo8Xf,C;YAAe,qBAAO,OA gxBH,OA hxBG,C;YAAP,uB;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBA+wBzB,OA/wByB,C;UAA5C,aiBjpnBO,W;UjBkpnBP,kBA8wBmB,O;UA7wBnB,iD;YAAgB,cAAhB,OB;YACl,cA4wBwB,SA5wBV,CAAU,WAAV,EAAuB,OAAvB,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;;QAywBP,yB;O;KafJ,C;4FAkBA,yB;MAzwBA,gD;MAAA,gE;MAywBA,gD;QAeW,6B;;UA1wBP,IA5hYO,qBAAQ,CA4hYf,C;YAAe,4BAAO,OA0wBI,OA1wBJ,C;YAAP,8B;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBAywBIB,OAzwBkB,C;UAA5C,aiBzqnBO,W;UjB0qnBP,kBAwwB0B,O;UAvwB1B,wD;YACl,cAswB+B,SA twBjB,CAAU,KAAV,EAAiB,WA AjB,EAA8B,UAAK,KAAL,CAA9B,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,M;;;QAmwBP,gC;O;KafJ,C;8FAkBA,yB;MANwBA,gD;MAAA,gE;MAmwBA,gD;QAgBW,6B;;UApwBP,IA7iYO,qBAAQ,CA6iYf,C;YAAe,4BAAO,OAowBI,OA pwBJ,C;YAAP,8B;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBAmwBIB,OA nwBkB,C;UAA5C,aiBlsnBO,W;UjBmsnBP,kBAkwB0B,O;UAjwB1B,wD;YACl,cAgwB+B,SAhwBjB,CAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,M;;;QA6vBP,gC;O;KAhBJ,C;8FamBA,yB;MA7vBA,gD;MAAA,gE;MA6vBA,gD;QAgBW,6B;;UA9vBP,IA9jYO,qBAAQ,CA8jYf,C;YAAe,4BAAO,OA8vBI,OA9vBJ,C;YAAP,8B;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBA6vBIB,OA7vBkB,C;UAA5C,aiB3tnBO,W;UjB4tnBP,kBA4vB0B,O;UA3vB1B,wD;YACl,cA0vB+B,SA1vBjB,CAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,M;;;QAuvBP,gC;O;KAhBJ,C;8FamBA,yB;MAvvBA,gD;MAAA,gE;MAuvBA,gD;QAgBW,6B;;UAxvBP,IAkYO,qBAAQ,CA+kYf,C;YAAe,4BAAO,OAwwBI,OA xvBJ,C;YAAP,8B;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBAuvBIB,OA vvBkB,C;UAA5C,aiBpvnBO,W;UjBqvnBP,kBASvB0B,O;UArvB1B,wD;YACl,cAovB+B,SApvBjB,CAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,M;;;QAivBP,gC;O;KAhBJ,C;8FamBA,yB;MAjvBA,gD;MAAA,gE;MAivBA,gD;Q

AgBW,6B;;UAlvBP,IAhmYO,qBAAQ,CAgmYf,C;YAAe,4BAAO,OAKvBI,OAlvBJ,C;YAAP,8B;WACqB,kBAA  
vB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBAivBIB,OAjvBkB,C;UAA5C,aiB7wnBO,W;UjB8wnBP,kBAgvB0B  
,O;UA/uB1B,wD;YACI,cA8uB+B,SA9uBjB,CAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;Y  
ACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,M;;;QA2uBP,gC;O;KAhBJ,C;8FamBA,yB;MA3uBA,gD;MAAA,g  
E;MA2uBA,gD;QAgBW,6B;;UA5uBP,IAjnYO,qBAAQ,CAinYf,C;YAAe,4BAAO,OA4uBI,OA5uBJ,C;YAAP,8B  
;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBA2uBIB,OA3uBkB,C;UAA5C,aiBtynBO,W;UjBuy  
nBP,kBA0uB0B,O;UAzuB1B,wD;YACI,cAwuB+B,SAxuBjB,CAAU,KAAV,EAAiB,WAAjB,EAA8B,UAAK,KA  
AL,CAA9B,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,M;;;QAquBP,gC;O;KAhBJ,C;8FamBA,yB;MARu  
BA,gD;MAAA,gE;MAquBA,gD;QAgBW,6B;;UAtuBP,IAloYO,qBAAQ,CAkoYf,C;YAAe,4BAAO,OA5uBI,OA  
tBJ,C;YAAP,8B;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBAquBIB,OAruBkB,C;UAA5C,aiB/  
znBO,W;UjBg0nBP,kBAouB0B,O;UAnuB1B,wD;YACI,cAkuB+B,SAluBjB,CAAU,KAAV,EAAiB,WAAjB,EAA  
8B,UAAK,KAAL,CAA9B,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,M;;;QA+tBP,gC;O;KAhBJ,C;8Fam  
BA,yB;MA/tBA,gD;MAAA,gE;MA+tBA,gD;QAgBW,6B;;UAhuBP,IANpYO,qBAAQ,CampYf,C;YAAe,4BAAO  
,OAguBI,OAhuBJ,C;YAAP,8B;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B,sBA+tBIB,OA/tBkB,C  
;UAA5C,aiBx1nBO,W;UjBy1nBP,kBA8tB0B,O;UA7tB1B,wD;YACI,cA4tB+B,SA5tBjB,CAAU,KAAV,EAAiB,  
WAAjB,EAA8B,UAAK,KAAL,CAA9B,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,M;;;QAYtBP,gC;O;K  
AhBJ,C;8FamBA,yB;MAztBA,gD;MAAA,gE;MAAA,oC;MAytBA,gD;QAgBW,6B;;UA1tBP,IApqYO,qBAAQ,C  
AoqYf,C;YAAe,4BAAO,OA0tBI,OA1tBJ,C;YAAP,8B;WACqB,kBAAvB,eAAa,mBAAO,CAAP,IAAb,C;UAA+B  
,sBAytBIB,OAztBkB,C;UAA5C,aiBj3nBO,W;UjBk3nBP,kBAwtB0B,O;UAvtB1B,wD;YACI,cAstB+B,SA  
ttBjB,CAAU,KAAV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,EAA9B,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,  
M;;;QAMtBP,gC;O;KAhBJ,C;gFamBA,+B;MAOoB,Q;MADhB,UAAe,C;MACf,wBAAgB,SAAhB,gB;QAAGB,c  
AAA,SAAhB,M;QACI,YAAO,SAAS,OAAT,CAAP,I;;MAEJ,OAAO,G;K;kFAGX,+B;MAOoB,Q;MADhB,UAAe  
,C;MACf,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,YAAO,SAAS,OAAT,CAAP,I;;MAEJ,OAAO,G;  
K;kFAGX,+B;MAOoB,Q;MADhB,UAAe,C;MACf,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,YAA  
O,SAAS,OAAT,CAAP,I;;MAEJ,OAAO,G;K;kFAGX,+B;MAOoB,Q;MADhB,UAAe,C;MACf,wBAAgB,SAAhB,  
gB;QAAGB,cAAA,SAAhB,M;QACI,YAAO,SAAS,OAAT,CAAP,I;;MAEJ,OAAO,G;K;kFAGX,+B;MAOoB,Q;M  
ADhB,UAAe,C;MACf,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,YAAO,SAAS,OAAT,CAAP,I;;MA  
EJ,OAAO,G;K;kFAGX,+B;MAOoB,Q;MADhB,UAAe,C;MACf,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;  
QACI,YAAO,SAAS,OAAT,CAAP,I;;MAEJ,OAAO,G;K;kFAGX,+B;MAOoB,Q;MADhB,UAAe,C;MACf,wBAA  
gB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,YAAO,SAAS,OAAT,CAAP,I;;MAEJ,OAAO,G;K;kFAGX,+B;M  
AOoB,Q;MADhB,UAAe,C;MACf,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,YAAO,SAAS,OAAT,C  
AAP,I;;MAEJ,OAAO,G;K;kFAGX,yB;MAAA,oC;MAAA,gC;MAAA,sC;QAOoB,Q;QADhB,UAAe,C;QACf,wB  
AAGB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UACI,YAAO,SAAS,oBAAT,CAAP,I;;QAEJ,OAAO,G;O;  
KAVX,C;4FAaA,+B;MAOoB,Q;MADhB,UAAkB,G;MACIB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;Q  
ACI,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;8FAGX,+B;MAOoB,Q;MADhB,UAAkB,G;MACIB,wBAAgB,  
SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;8FAGX,+B;MAOoB,Q  
;MADhB,UAAkB,G;MACIB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,OAAO,SAAS,OAAT,C;;MA  
EX,OAAO,G;K;8FAGX,+B;MAOoB,Q;MADhB,UAAkB,G;MACIB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAh  
B,M;QACI,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;8FAGX,+B;MAOoB,Q;MADhB,UAAkB,G;MACIB,wB  
AAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;8FAGX,+B;MA  
OoB,Q;MADhB,UAAkB,G;MACIB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,OAAO,SAAS,OAAT,  
C;;MAEX,OAAO,G;K;8FAGX,+B;MAOoB,Q;MADhB,UAAkB,G;MACIB,wBAAgB,SAAhB,gB;QAAGB,cAAA,  
SAAhB,M;QACI,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;8FAGX,+B;MAOoB,Q;MADhB,UAAkB,G;MACI  
B,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;8FAGX,y  
B;MAAA,oC;MAAA,gC;MAAA,sC;QAOoB,Q;QADhB,UAAkB,G;QACIB,wBAAgB,SAAhB,gB;UAAgB,cAAh  
B,UAAgB,SAAhB,O;UACI,OAAO,SAAS,oBAAT,C;;QAEJ,OAAO,G;O;KAVX,C;gFAaA,+B;MAUoB,Q;MADh  
B,UAAoB,C;MACpB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,OAAO,SAAS,OAAT,C;;MAEX,O  
AAO,G;K;kFAGX,+B;MAUoB,Q;MADhB,UAAoB,C;MACpB,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;



gB;UAAgB,cAAA,SAAhB,M;UACI,MGzyqBiD,cHyyqBjD,GGzyqB2D,KAAK,GHyyqBzD,SAAS,OAAT,CGzyqBoE,KAAx,IAAf,C;;QH2yqBrD,OAAO,G;O;KAdX,C;mFAiBA,yB;MGt/pBA,6B;MHs/pBA,sC;QAWoB,Q;QADhB,UGt/pBmC,cHs/pBnB,CGt/pBmB,C;QH/pBnC,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,MG1zqBiD,cH0zqBjD,GG1zqB2D,KAAK,GH0zqBzD,SAAS,OAAT,CG1zqBoE,KAAx,IAAf,C;;QH4zqBrD,OAAO,G;O;KAdX,C;mFAiBA,yB;MGvgqBA,6B;MHugqBA,sC;QAWoB,Q;QADhB,UGvgqBmC,cHugqBnB,CGvgqBmB,C;QHwgqBnB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,MG30qBiD,cH20qBjD,GG30qB2D,KAAK,GH20qBzD,SAAS,OAAT,CG30qBoE,KAAx,IAAf,C;;QH60qBrD,OAAO,G;O;KAdX,C;mFAiBA,yB;MGxhqBA,6B;MHwhqBA,sC;QAWoB,Q;QADhB,UGxhqBmC,cHwhqBnB,CGxhqBmB,C;QHxhqBnB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,MG51qBiD,cH41qBjD,GG51qB2D,KAAK,GH41qBzD,SAAS,OAAT,CG51qBoE,KAAx,IAAf,C;;QH81qBrD,OAAO,G;O;KAdX,C;mFAiBA,yB;MGziqBA,6B;MHyiqBA,sC;QAWoB,Q;QADhB,UGziqBmC,cHyiqBnB,CGziqBmB,C;QH0iqBnB,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,MG72qBiD,cH62qBjD,GG72qB2D,KAAK,GH62qBzD,SAAS,OAAT,CG72qBoE,KAAx,IAAf,C;;QH+2qBrD,OAAO,G;O;KAdX,C;mFAiBA,yB;MAAA,oC;MAAA,gC;MG1jqBA,6B;MH0jqBA,sC;QAWoB,Q;QADhB,UG1jqBmC,cH0jqBnB,CG1jqBmB,C;QH2jqBnB,wBAAgB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UACI,MG93qBiD,cH83qBjD,GG93qB2D,KAAK,GH83qBzD,SAAS,oBAAT,CG93qBoE,KAAx,IAAf,C;;QHg4qBrD,OAAO,G;O;KAdX,C;mFAiBA,yB;MmBxkqBA,+B;MnBwkqBA,sC;QAWoB,Q;QADhB,UmBvkqBqC,eAAW,oBnBukqB/B,CmBvkqB+B,CAAX,C;QnBwkqBrC,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,MmB54qBmD,enB44qBnD,GmB54qB8D,KAAK,KnB44qB5D,SAAS,OAAT,CmB54qBuE,KAAx,CAAhB,C;;QnB84qBvD,OAAO,G;O;KAdX,C;mFAiBA,yB;MmBzqlqBA,+B;MnBylqBA,sC;QAWoB,Q;QADhB,UmBxlqBqC,eAAW,oBnBwlqB/B,CmBxlqB+B,CAAX,C;QnBylqBrC,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,MmB75qBmD,enB65qBnD,GmB75qB8D,KAAK,KnB65qB5D,SAAS,OAAT,CmB75qBuE,KAAx,CAAhB,C;;QnB+5qBvD,OAAO,G;O;KAdX,C;mFAiBA,yB;MmB1mqBA,+B;MnB0mqBA,sC;QAWoB,Q;QADhB,UmBzmqBqC,eAAW,oBnBmqB/B,CmBzmqB+B,CAAX,C;QnB0mqBrC,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,MmB96qBmD,enB86qBnD,GmB96qB8D,KAAK,KnB86qB5D,SAAS,OAAT,CmB96qBuE,KAAx,CAAhB,C;;QnBg7qBvD,OAAO,G;O;KAdX,C;kFAiBA,yB;MmB3nqBA,+B;MnB2nqBA,sC;QAWoB,Q;QADhB,UmB1nqBqC,eAAW,oBnB0nqB/B,CmB1nqB+B,CAAX,C;QnB2nqBrC,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,MmB/7qBmD,enB+7qBnD,GmB/7qB8D,KAAK,KnB+7qB5D,SAAS,OAAT,CmB/7qBuE,KAAx,CAAhB,C;;QnBi8qBvD,OAAO,G;O;KAdX,C;mFAiBA,yB;MmB5oqBA,+B;MnB4oqBA,sC;QAWoB,Q;QADhB,UmB3oqBqC,eAAW,oBnB2oqB/B,CmB3oqB+B,CAAX,C;QnB4oqBrC,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,MmBh9qBmD,enBg9qBnD,GmBh9qB8D,KAAK,KnBg9qB5D,SAAS,OAAT,CmBh9qBuE,KAAx,CAAhB,C;;QnBk9qBvD,OAAO,G;O;KAdX,C;mFAiBA,yB;MmB7pqBA,+B;MnB6pqBA,sC;QAWoB,Q;QADhB,UmB5pqBqC,eAAW,oBnB4pqB/B,CmB5pqB+B,CAAX,C;QnB6pqBrC,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,MmBj+qBmD,enBi+qBnD,GmBj+qB8D,KAAK,KnBi+qB5D,SAAS,OAAT,CmBj+qBuE,KAAx,CAAhB,C;;QnBm+qBvD,OAAO,G;O;KAdX,C;mFAiBA,yB;MmB9qqBA,+B;MnB8qqBA,sC;QAWoB,Q;QADhB,UmB7qqBqC,eAAW,oBnB6qqB/B,CmB7qqB+B,CAAX,C;QnB8qqBrC,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,MmBl/qBmD,enBk/qBnD,GmBl/qB8D,KAAK,KnBk/qB5D,SAAS,OAAT,CmBl/qBuE,KAAx,CAAhB,C;;QnBo/qBvD,OAAO,G;O;KAdX,C;kFAiBA,yB;MmB/rqBA,+B;MnB+rqBA,sC;QAWoB,Q;QADhB,UmB9rqBqC,eAAW,oBnB8rqB/B,CmB9rqB+B,CAAX,C;QnB+rqBrC,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,MmBngrBmD,enBmgrBnD,GmBngrB8D,KAAK,KnBmgrB5D,SAAS,OAAT,CmBngrBuE,KAAx,CAAhB,C;;QnBqgrBvD,OAAO,G;O;KAdX,C;mFAiBA,yB;MAAA,oC;MAAA,gC;MmBhtqBA,+B;MnBgtqBA,sC;QAWoB,Q;QADhB,UmB/sqBqC,eAAW,oBnB+sqB/B,CmB/sqB+B,CAAX,C;QnBgtqBrC,wBAAgB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UACI,MmBphrBmD,enBohrBnD,GmBphrB8D,KAAK,KnBohrB5D,SAAS,oBAAT,CmBphrBuE,KAAx,CAAhB,C;;QnBshrBvD,OAAO,G;O;KAdX,C;IAiBA,mC;MAIoB,UAMT,M;MANP,wBAAgB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,IAAI,eAAJ,C;UACI,MAAM,gCAAYB,2BAAwB,SAAxB,MAAzB,C;;MAId,OAAO,0D;K;wFAGX,yB;MAAA,+D;MAAA,6B;MAAA,uC;QAUoB,Q;QAFhB,YAAY,gB;QACZ,aAAa,gB;QACb,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,IAAI,UAAU,OAAV,CAAJ,C;YACI,KAAM,WAAI,OAAJ,C;;YAEN,MAAO,WAAI,OAAJ,C;;QAGf,OAAO,cAAK,KAAL,EAAY,MAAZ,C;O;KAjBX,C;0FAoBA,yB;MAAA,+D;MAAA,6B;MAAA,uC;QAUoB,Q;QAFhB,YAAY,gB;QACZ,aAAa,gB;QACb,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI

,IAAI,UAAU,OAAV,CAAJ,C;YACI,KAAM,WAAI,OAAJ,C;;YAEN,MAAO,WAAI,OAAJ,C;;;QAGf,OAAO,cA  
AK,KAAL,EAAY,MAAZ,C;O;KAjBX,C;0FAoBA,yB;MAAA,+D;MAAA,6B;MAAA,uC;QAUoB,Q;QAFhB,YA  
AY,gB;QACZ,aAAa,gB;QACb,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,IAAI,UAAU,OAAV,CAA  
J,C;YACI,KAAM,WAAI,OAAJ,C;;YAEN,MAAO,WAAI,OAAJ,C;;;QAGf,OAAO,cAAK,KAAL,EAAY,MAAZ,  
C;O;KAjBX,C;0FAoBA,yB;MAAA,+D;MAAA,6B;MAAA,uC;QAUoB,Q;QAFhB,YAAY,gB;QACZ,aAAa,gB;Q  
ACb,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,IAAI,UAAU,OAAV,CAAJ,C;YACI,KAAM,WAAI,  
OAAJ,C;;YAEN,MAAO,WAAI,OAAJ,C;;;QAGf,OAAO,cAAK,KAAL,EAAY,MAAZ,C;O;KAjBX,C;0FAoBA,y  
B;MAAA,+D;MAAA,6B;MAAA,uC;QAUoB,Q;QAFhB,YAAY,gB;QACZ,aAAa,gB;QACb,wBAAgB,SAAhB,gB  
;UAAgB,cAAA,SAAhB,M;UACI,IAAI,UAAU,OAAV,CAAJ,C;YACI,KAAM,WAAI,OAAJ,C;;YAEN,MAAO,W  
AAI,OAAJ,C;;;QAGf,OAAO,cAAK,KAAL,EAAY,MAAZ,C;O;KAjBX,C;0FAoBA,yB;MAAA,+D;MAAA,6B;M  
AAA,uC;QAUoB,Q;QAFhB,YAAY,gB;QACZ,aAAa,gB;QACb,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;  
UACI,IAAI,UAAU,OAAV,CAAJ,C;YACI,KAAM,WAAI,OAAJ,C;;YAEN,MAAO,WAAI,OAAJ,C;;;QAGf,OAA  
O,cAAK,KAAL,EAAY,MAAZ,C;O;KAjBX,C;0FAoBA,yB;MAAA,+D;MAAA,6B;MAAA,uC;QAUoB,Q;QAFh  
B,YAAY,gB;QACZ,aAAa,gB;QACb,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,IAAI,UAAU,OAAV  
,CAAJ,C;YACI,KAAM,WAAI,OAAJ,C;;YAEN,MAAO,WAAI,OAAJ,C;;;QAGf,OAAO,cAAK,KAAL,EAAY,M  
AAZ,C;O;KAjBX,C;0FAoBA,yB;MAAA,+D;MAAA,6B;MAAA,uC;QAUoB,Q;QAFhB,YAAY,gB;QACZ,aAAa,  
gB;QACb,wBAAgB,SAAhB,gB;UAAgB,cAAA,SAAhB,M;UACI,IAAI,UAAU,OAAV,CAAJ,C;YACI,KAAM,W  
AAI,OAAJ,C;;YAEN,MAAO,WAAI,OAAJ,C;;;QAGf,OAAO,cAAK,KAAL,EAAY,MAAZ,C;O;KAjBX,C;0FAo  
BA,yB;MAAA,+D;MAAA,oC;MAAA,gC;MAAA,6B;MAAA,uC;QAUoB,Q;QAFhB,YAAY,gB;QACZ,aAAa,gB;  
QACb,wBAAgB,SAAhB,gB;UAAgB,cAAhB,UAAgB,SAAhB,O;UACI,IAAI,UAAU,oBAAV,CAAJ,C;YACI,KA  
AM,WAAI,oBAAJ,C;;YAEN,MAAO,WAAI,oBAAJ,C;;;QAGf,OAAO,cAAK,KAAL,EAAY,MAAZ,C;O;KAjBX,  
C;IAoBA,+B;MAkGI,WkB3orBO,MAAO,KIB2orBG,gBkB3orBH,EIBgjrBH,KA2FkB,OkB3orBf,C;MIB4orBd,W  
AAW,iBAAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WA9FqB,GA8FP,UAAK,CAAL,CA  
9FO,EAAnB,KA8FqB,CAAM,CAAN,CA9FF,CA8FrB,C;;MA9FT,OAGGO,I;K;IA7FX,iC;MAwGI,WkB3prBO,M  
AAO,KIB2prBG,gBkB3prBH,EIB0jrBH,KAIgkB,OkB3prBf,C;MIB4prBd,WAAW,iBAAa,IAAb,C;MACX,aAAU,  
CAAV,MAAkB,IAAIB,M;QACI,IAAK,WApGqB,GAoGP,UAAK,CAAL,CAPGO,EAAnB,KAoGqB,CAAM,CA  
AN,CAPGF,CAoGrB,C;;MApGT,OAsGO,I;K;IAngX,iC;MA8GI,WkB3qrBO,MAAO,KIB2qrBG,gBkB3qrBH,El  
BokrBH,KAuGkB,OkB3qrBf,C;MIB4qrBd,WAAW,iBAAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QA  
CI,IAAK,WA1GqB,GA0GP,UAAK,CAAL,CA1GO,EAAnB,KA0GqB,CAAM,CAAN,CA1GF,CA0GrB,C;;MA1G  
T,OA4GO,I;K;IAzGX,iC;MAoHI,WkB3rrBO,MAAO,KIB2rrBG,gBkB3rrBH,EIB8krBH,KA6GkB,OkB3rrBf,C;M  
IB4rrBd,WAAW,iBAAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WAhHqB,GAgHP,UAA  
K,CAAL,CAhHO,EAAnB,KAgHqB,CAAM,CAAN,CAhHF,CAGhrB,C;;MAhHT,OAKHO,I;K;IA/GX,iC;MA0HI,  
WkB3srBO,MAAO,KIB2srBG,gBkB3srBH,EIBwlrBH,KAmHkB,OkB3srBf,C;MIB4srBd,WAAW,iBAAa,IAAb,C  
;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WAtHqB,GAsHP,UAAK,CAAL,CAtHO,EAAnB,KAsHq  
B,CAAM,CAAN,CAtHF,CAsHrB,C;;MAtHT,OAwHO,I;K;IARHX,iC;MAGII,WkB3trBO,MAAO,KIB2trBG,gBkB  
3trBH,EIBkMrBH,KAYHkB,OkB3trBf,C;MIB4trBd,WAAW,iBAAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAA  
IB,M;QACI,IAAK,WA5HqB,GA4HP,UAAK,CAAL,CA5HO,EAAnB,KA4HqB,CAAM,CAAN,CA5HF,CA4HrB,  
C;;MA5HT,OA8HO,I;K;IA3HX,iC;MA5II,WkB3urBO,MAAO,KIB2urBG,gBkB3urBH,EIB4mrBH,KA+HkB,OkB  
3urBf,C;MIB4urBd,WAAW,iBAAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WAlIqB,GAKI  
P,UAAK,CAAL,CAlIO,EAAnB,KAKIqB,CAAM,CAAN,CAlIF,CAKIrB,C;;MAIIT,OAoIO,I;K;IAjIX,iC;MA4II,W  
kB3vrBO,MAAO,KIB2vrBG,gBkB3vrBH,EIBsnrBH,KAQIkB,OkB3vrBf,C;MIB4vrBd,WAAW,iBAAa,IAAb,C;M  
ACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WAXIqB,GAwIP,UAAK,CAAL,CAXIO,EAAnB,KAwIqB,C  
AAM,CAAN,CAXIF,CAwIrB,C;;MAXIT,OAoIO,I;K;IAvIX,iC;MAkJI,WkB3wrBO,MAAO,KIB2wrBG,gBkB3wr  
BH,EIBgorBH,KA2IkB,OkB3wrBf,C;MIB4wrBd,WAAW,iBAAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,  
M;QACI,IAAK,WA9IqB,GA8IP,sBAAK,CAAL,EA9IO,EAAnB,KA8IqB,CAAM,CAAN,CA9IF,CA8IrB,C;;MA9  
IT,OAGJO,I;K;8EA7IX,yB;MAAA,gE;MkBzorBA,iB;MIByorBA,8C;QAQI,WkB3orBO,MAAO,KIB2orBG,gBkB  
3orBH,EIB2orBS,KAAM,OkB3orBf,C;QIB4orBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M  
;UACI,IAAK,WAAI,UAAU,UAAK,CAAL,CAAV,EAAM,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;

KAbX,C;8EAgBA,yB;MAAA,gE;MkBzprBA,iB;MIByprBA,8C;QAQI,WkB3prBO,MAAO,KIB2prBG,gBkB3prB  
H,EIB2prBS,KAAM,OkB3prBf,C;QIB4prBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UA  
CI,IAAK,WAAI,UAAU,UAAK,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAb  
X,C;+EAgBA,yB;MAAA,gE;MkBzqrBA,iB;MIByqrBA,8C;QAQI,WkB3qrBO,MAAO,KIB2qrBG,gBkB3qrBH,El  
B2qrBS,KAAM,OkB3qrBf,C;QIB4qrBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,I  
AAK,WAAI,UAAU,UAAK,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C  
;8EAgBA,yB;MAAA,gE;MkBzrrBA,iB;MIByrrBA,8C;QAQI,WkB3rrBO,MAAO,KIB2rrBG,gBkB3rrBH,EIB2rrB  
S,KAAM,OkB3rrBf,C;QIB4rrBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,W  
AAI,UAAU,UAAK,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;+EAgB  
A,yB;MAAA,gE;MkBzsrBA,iB;MIBysrBA,8C;QAQI,WkB3srBO,MAAO,KIB2srBG,gBkB3srBH,EIB2srBS,KAA  
M,OkB3srBf,C;QIB4srBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,U  
AAU,UAAK,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;+EAgBA,yB;  
MAAA,gE;MkBztrBA,iB;MIBytrBA,8C;QAQI,WkB3trBO,MAAO,KIB2trBG,gBkB3trBH,EIB2trBS,KAAM,OkB  
3trBf,C;QIB4trBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,U  
AAK,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;+EAgBA,yB;MAAA,  
gE;MkBzurBA,iB;MIByurBA,8C;QAQI,WkB3urBO,MAAO,KIB2urBG,gBkB3urBH,EIB2urBS,KAAM,OkB3urB  
f,C;QIB4urBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,UAA  
K,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;+EAgBA,yB;MAAA,gE;  
MkBzvrBA,iB;MIByvrBA,8C;QAQI,WkB3vrBO,MAAO,KIB2vrBG,gBkB3vrBH,EIB2vrBS,KAAM,OkB3vrBf,C;  
QIB4vrBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,UAAK,C  
AAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;+EAgBA,yB;MAAA,gE;MA  
AA,oC;MkBzwrBA,iB;MIBywrBA,8C;QAQI,WkB3wrBO,MAAO,KIB2wrBG,gBkB3wrBH,EIB2wrBS,KAAM,O  
kB3wrBf,C;QIB4wrBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAA  
U,sBAAK,CAAL,EA AV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;IAgBA,kC;MAq  
GoB,gB;MAHhB,gBAAGB,gB;MACHB,WAAW,iBkBt3rBJ,MAAO,KIBs3rBsB,wBA5FzB,KA4FyB,EAAwB,EA  
AxB,CkBt3rBtB,EIBs3rBmD,SkBt3rBnD,CIBs3rBH,C;MACX,QAAQ,C;MACQ,OA9FL,KA8FK,W;MAAhB,OA  
AgB,cAAhB,C;QAAGB,yB;QACZ,IAAI,KA AK,SAAT,C;UAAoB,K;QACpB,IAAK,WAhGqB,GAGGP,UA AK,U  
AAL,EAAK,kBAAL,SAhGO,EAAGI,OA hGJ,CAGrB,C;;MAhGT,OakGO,I;K;IA/FX,kC;MA6GoB,gB;MAHhB,  
gBAAGB,gB;MACHB,WAAW,iBkBx4rBJ,MAAO,KIBw4rBsB,wBApGzB,KAoGyB,EAAwB,EA AxB,CkBx4rBtB  
,EIBw4rBmD,SkBx4rBnD,CIBw4rBH,C;MACX,QAAQ,C;MACQ,OA tGL,KAsGK,W;MAAhB,OAAGB,cAAhB,C  
;QAAGB,yB;QACZ,IAAI,KA AK,SAAT,C;UAAoB,K;QACpB,IAAK,WAxGqB,GAwGP,UA AK,UAAL,EAAK,kB  
AAL,SAxGO,EA wGI,OA xGJ,CAwGrB,C;;MAxGT,OA0GO,I;K;IAvGX,kC;MAqHoB,gB;MAHhB,gBAAGB,gB;  
MACHB,WAAW,iBkB15rBJ,MAAO,KIB05rBsB,wBA5GzB,KA4GyB,EAAwB,EA AxB,CkB15rBtB,EIB05rBmD,  
SkB15rBnD,CIB05rBH,C;MACX,QAAQ,C;MACQ,OA9GL,KA8GK,W;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;  
QACZ,IAAI,KA AK,SAAT,C;UAAoB,K;QACpB,IAAK,WAhHqB,GAGHP,UA AK,UAAL,EAAK,kBAAL,SAhHO  
,EA gHI,OA hHJ,CAGrB,C;;MAhHT,OakHO,I;K;IA/GX,kC;MA6HoB,gB;MAHhB,gBAAGB,gB;MACHB,WAA  
W,iBkB56rBJ,MAAO,KIB46rBsB,wBApHzB,KAoHyB,EAAwB,EA AxB,CkB56rBtB,EIB46rBmD,SkB56rBnD,Cl  
B46rBH,C;MACX,QAAQ,C;MACQ,OA tHL,KAsHK,W;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,IAAI,KA  
AK,SAAT,C;UAAoB,K;QACpB,IAAK,WAxHqB,GAwHP,UA AK,UAAL,EAAK,kBAAL,SAxHO,EA wHI,OA xH  
J,CAwHrB,C;;MAxHT,OA0HO,I;K;IAvHX,kC;MAqIoB,gB;MAHhB,gBAAGB,gB;MACHB,WAAW,iBkB97rBJ,  
MAAO,KIB87rBsB,wBA5HzB,KA4HyB,EAAwB,EA AxB,CkB97rBtB,EIB87rBmD,SkB97rBnD,CIB87rBH,C;M  
ACX,QAAQ,C;MACQ,OA9HL,KA8HK,W;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,IAAI,KA AK,SAAT,C  
;UAAoB,K;QACpB,IAAK,WAhIqB,GAGIP,UA AK,UAAL,EAAK,kBAAL,SAhIO,EA gII,OA hIJ,CAGrB,C;;MAhI  
T,OakIO,I;K;IA/HX,kC;MA6IoB,gB;MAHhB,gBAAGB,gB;MACHB,WAAW,iBkBh9rBJ,MAAO,KIBg9rBsB,wB  
ApIzB,KAoIyB,EAAwB,EA AxB,CkBh9rBtB,EIBg9rBmD,SkBh9rBnD,CIBg9rBH,C;MACX,QAAQ,C;MACQ,O  
AtIL,KAsIK,W;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,IAAI,KA AK,SAAT,C;UAAoB,K;QACpB,IAAK,  
WAxIqB,GAwIP,UA AK,UAAL,EAAK,kBAAL,SAxIO,EA wII,OA xIJ,CAwIrB,C;;MAxIT,OA0IO,I;K;IAvIX,kC;  
MAqJoB,gB;MAHhB,gBAAGB,gB;MACHB,WAAW,iBkB1+rBJ,MAAO,KIBk+rBsB,wBA5IzB,KA4IyB,EAAwB,

EAAxB,CkBl+rBtB,ElBk+rBmD,SkBl+rBnD,CIBk+rBH,C;MACX,QAAQ,C;MACQ,OA9IL,KA8IK,W;MAAhB, OAAgB,cAAhB,C;QAAgB,yB;QACZ,IAAI,KAAK,SAAT,C;UAAoB,K;QACpB,IAAK,WAhJqB,GAgJP,UAAK, UAAL,EAAK,kBAAL,SAhJO,EAgJI,OAhJJ,CAGJrB,C;;MAhJT,OAKJO,I;K;IA/IX,kC;MA6JoB,gB;MAHhB,gBA AgB,gB;MACHB,WAAW,iBkBp/rBJ,MAAO,KIBo/rBsB,wBApJzB,KAOjyB,EAAwB,EAAxB,CkBp/rBtB,ElBo/r BmD,SkBp/rBnD,CIBo/rBH,C;MACX,QAAQ,C;MACQ,OAtJL,KAsJK,W;MAAhB,OAAgB,cAAhB,C;QAAgB,y B;QACZ,IAAI,KAAK,SAAT,C;UAAoB,K;QACpB,IAAK,WAxJqB,GAwJP,UAAK,UAAL,EAAK,kBAAL,SAxJ O,EAwJI,OAxJJ,CAwJrB,C;;MAxJT,OA0JO,I;K;IAvJX,kC;MAqKoB,gB;MAHhB,gBAAgB,gB;MACHB,WAAW, iBkbtgsBJ,MAAO,KIBsgsBsB,wBA5JzB,KA4JyB,EAAwB,EAAxB,CkbtgsBtB,ElBsgsBmD,SkbtgsBnD,CIBsgsB H,C;MACX,QAAQ,C;MACQ,OA9JL,KA8JK,W;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,IAAI,KAAK,SA AT,C;UAAoB,K;QACpB,IAAK,WAhKqB,GAgKP,sBAAK,UAAL,EAAK,kBAAL,UAhKO,EAgKI,OAhKJ,CAG KrB,C;;MAhKT,OAKKO,I;K;+EA/JX,yB;MAAA,kF;MAAA,gE;Mkbn3rBA,iB;MIBm3rBA,8C;QAWoB,UAEY, M;QAL5B,gBAAgB,gB;QACHB,WAAW,ekBt3rBJ,MAAO,KIBs3rBsB,wBAAN,KAAM,EAAwB,EAAxB,Ckbt3r BtB,ElBs3rBmD,Skbt3rBnD,CIBs3rBH,C;QACX,QAAQ,C;QACQ,uB;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;U ACZ,IAAI,KAAK,SAAT,C;YAAoB,K;UACpB,IAAK,WAAI,UAAU,UAAK,UAAL,EAAK,kBAAL,SAAV,EAAq B,OAARb,CAAJ,C;;QAET,OAAO,I;O;KafX,C;+EakBA,yB;MAAA,kF;MAAA,gE;MkBr4rBA,iB;MIBq4rBA,8C ;QAWoB,UAEY,M;QAL5B,gBAAgB,gB;QACHB,WAAW,ekBx4rBJ,MAAO,KIBw4rBsB,wBAAN,KAAM,EAA wB,EAAxB,CkBx4rBtB,ElBw4rBmD,SkBx4rBnD,CIBw4rBH,C;QACX,QAAQ,C;QACQ,uB;QAAhB,OAAgB,cA AhB,C;UAAgB,yB;UACZ,IAAI,KAAK,SAAT,C;YAAoB,K;UACpB,IAAK,WAAI,UAAU,UAAK,UAAL,EAAK, kBAAL,SAAV,EAAqB,OAARb,CAAJ,C;;QAET,OAAO,I;O;KafX,C;+EakBA,yB;MAAA,kF;MAAA,gE;MkBv5 rBA,iB;MIBu5rBA,8C;QAWoB,UAEY,M;QAL5B,gBAAgB,gB;QACHB,WAAW,ekB15rBJ,MAAO,KIB05rBsB,w BAAN,KAAM,EAAwB,EAAxB,CkB15rBtB,ElB05rBmD,SkB15rBnD,CIB05rBH,C;QACX,QAAQ,C;QACQ,uB; QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,IAAI,KAAK,SAAT,C;YAAoB,K;UACpB,IAAK,WAAI,UAAU,U AAK,UAAL,EAAK,kBAAL,SAAV,EAAqB,OAARb,CAAJ,C;;QAET,OAAO,I;O;KafX,C;+EakBA,yB;MAAA,k F;MAAA,gE;MkBz6rBA,iB;MIBy6rBA,8C;QAWoB,UAEY,M;QAL5B,gBAAgB,gB;QACHB,WAAW,ekB56rBJ, MAAO,KIB46rBsB,wBAAN,KAAM,EAAwB,EAAxB,CkB56rBtB,ElB46rBmD,SkB56rBnD,CIB46rBH,C;QACX, QAAQ,C;QACQ,uB;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,IAAI,KAAK,SAAT,C;YAAoB,K;UACpB,IA AK,WAAI,UAAU,UAAK,UAAL,EAAK,kBAAL,SAAV,EAAqB,OAARb,CAAJ,C;;QAET,OAAO,I;O;KafX,C;+ EakBA,yB;MAAA,kF;MAAA,gE;MkB37rBA,iB;MIB27rBA,8C;QAWoB,UAEY,M;QAL5B,gBAAgB,gB;QACH B,WAAW,ekB97rBJ,MAAO,KIB87rBsB,wBAAN,KAAM,EAAwB,EAAxB,CkB97rBtB,ElB87rBmD,SkB97rBnD ,CIB87rBH,C;QACX,QAAQ,C;QACQ,uB;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,IAAI,KAAK,SAAT,C; YAAoB,K;UACpB,IAAK,WAAI,UAAU,UAAK,UAAL,EAAK,kBAAL,SAAV,EAAqB,OAARb,CAAJ,C;;QAET, OAAO,I;O;KafX,C;+EakBA,yB;MAAA,kF;MAAA,gE;MkB78rBA,iB;MIB68rBA,8C;QAWoB,UAEY,M;QAL5 B,gBAAgB,gB;QACHB,WAAW,ekBh9rBJ,MAAO,KIBg9rBsB,wBAAN,KAAM,EAAwB,EAAxB,CkBh9rBtB,El Bg9rBmD,SkBh9rBnD,CIBg9rBH,C;QACX,QAAQ,C;QACQ,uB;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,I AAI,KAAK,SAAT,C;YAAoB,K;UACpB,IAAK,WAAI,UAAU,UAAK,UAAL,EAAK,kBAAL,SAAV,EAAqB,OA ARb,CAAJ,C;;QAET,OAAO,I;O;KafX,C;+EakBA,yB;MAAA,kF;MAAA,gE;MkB/9rBA,iB;MIB+9rBA,8C;QA WoB,UAEY,M;QAL5B,gBAAgB,gB;QACHB,WAAW,ekBl+rBJ,MAAO,KIBk+rBsB,wBAAN,KAAM,EAAwB,E AAXB,CkBl+rBtB,ElBk+rBmD,SkBl+rBnD,CIBk+rBH,C;QACX,QAAQ,C;QACQ,uB;QAAhB,OAAgB,cAAhB,C ;UAAgB,yB;UACZ,IAAI,KAAK,SAAT,C;YAAoB,K;UACpB,IAAK,WAAI,UAAU,UAAK,UAAL,EAAK,kBAA L,SAAV,EAAqB,OAARb,CAAJ,C;;QAET,OAAO,I;O;KafX,C;+EakBA,yB;MAAA,kF;MAAA,gE;MkBj/rBA,iB; MIBi/rBA,8C;QAWoB,UAEY,M;QAL5B,gBAAgB,gB;QACHB,WAAW,ekBp/rBJ,MAAO,KIBo/rBsB,wBAAN,K AAM,EAAwB,EAAxB,CkBp/rBtB,ElBo/rBmD,SkBp/rBnD,CIBo/rBH,C;QACX,QAAQ,C;QACQ,uB;QAAhB,OA AgB,cAAhB,C;UAAgB,yB;UACZ,IAAI,KAAK,SAAT,C;YAAoB,K;UACpB,IAAK,WAAI,UAAU,UAAK,UAAL ,EAAK,kBAAL,SAAV,EAAqB,OAARb,CAAJ,C;;QAET,OAAO,I;O;KafX,C;+EakBA,yB;MAAA,kF;MAAA,gE; MAAA,oC;MkBngsBA,iB;MIBmgsBA,8C;QAWoB,UAEY,M;QAL5B,gBAAgB,gB;QACHB,WAAW,ekBtgsBJ,M AAO,KIBsgsBsB,wBAAN,KAAM,EAAwB,EAAxB,CkbtgsBtB,ElBsgsBmD,SkbtgsBnD,CIBsgsBH,C;QACX,QA AQ,C;QACQ,uB;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,IAAI,KAAK,SAAT,C;YAAoB,K;UACpB,IAAK, WAAI,UAAU,sBAAK,UAAL,EAAK,kBAAL,UAAV,EAAqB,OAARb,CAAJ,C;;QAET,OAAO,I;O;KafX,C;IAkB

A,kC;MAwFI,WkBvmsBO,MAAO,KIBumsBG,gBkBvmsBH,ElBshsBH,KAiFkB,OkBvmsBf,C;MIBwmsBd,WAAW,iBAaAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WApFqB,GAoFP,UAAK,CAAL,CAPFO,EAAnB,KAOFqB,CAAM,CAAN,CAPFF,CAoFrB,C;;MAPFT,OASFO,I;K;IANFX,kC;MA8FI,WkBvnsBO,MAAO,KIBunsBG,gBkBvnsBH,ElBgisBH,KAuFkB,OkBvnsBf,C;MIBwnsBd,WAAW,iBAaAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WA1FqB,GA0FP,UAAK,CAAL,CA1FO,EAAnB,KAOFqB,CAAM,CAAN,CA1FF,CA0FrB,C;;MA1FT,OA4FO,I;K;IAZFX,kC;MAoGI,WkBvosBO,MAAO,KIBuosBG,gBkBvosBH,ElB0isBH,KA6FkB,OkBvosBf,C;MIBwosBd,WAAW,iBAaAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WAhGqB,GAgGP,UAAK,CAAL,CAhGO,EAAnB,KAgGqB,CAAM,CAAN,CAhGF,CAGrB,C;;MAhGT,OAkGO,I;K;IA/FX,kC;MA0GI,WkBvpsBO,MAAO,KIBupsBG,gBkBvpsBH,ElBojsBH,KAmGkB,OkBvpsBf,C;MIBwpsBd,WAAW,iBAaAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WAtGqB,GAsGP,UAAK,CAAL,CAtGO,EAAnB,KAsGqB,CAAM,CAAN,CAtGF,CAsGrB,C;;MATGT,OAwoGO,I;K;IARGX,kC;MAGHI,WkBvqsBO,MAAO,KIBuqsBG,gBkBvqsBH,ElB8jsBH,KAyGkB,OkBvqsBf,C;MIBwqsBd,WAAW,iBAaAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WA5GqB,GA4GP,UAAK,CAAL,CA5GO,EAAnB,KA4GqB,CAAM,CAAN,CA5GF,CA4GrB,C;;MA5GT,OA8GO,I;K;IA3GX,kC;MAsHI,WkBvrsBO,MAAO,KIBursBG,gBkBvrsBH,ElBwksBH,KA+GkB,OkBvrsBf,C;MIBwrsBd,WAAW,iBAaAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WAlHqB,GakHP,UAAK,CAAL,CAIHO,EAAnB,KAKHqB,CAAM,CAAN,CAIHF,CakHrB,C;;MAIHT,OAoHO,I;K;IAjHX,kC;MA4HI,WkBvssBO,MAAO,KIBussBG,gBkBvssBH,ElBklsBH,KaqHkB,OkBvssBf,C;MIBwssBd,WAAW,iBAaAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WAxHqB,GawHP,UAAK,CAAL,CAXHO,EAAnB,KAwHqB,CAAM,CAAN,CAXHF,CAwHrB,C;;MAxHT,OA0HO,I;K;IAvHX,kC;MAkII,WkBvtsBO,MAAO,KIButsBG,gBkBvtsBH,ElB4lsBH,KA2HkB,OkBvtsBf,C;MIBwtsBd,WAAW,iBAaAa,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WA9HqB,GA8HP,sBAAK,CAAL,EA9HO,EA8HE,YA9HrB,KA8HqB,CAAM,CAAN,EA9HF,CA8HrB,C;;MA9HT,OAgIO,I;K;+EA7HX,yB;MAAA,gE;MkBrmsBA,iB;MIBqmsBA,8C;QAQI,WkBvmsBO,MAAO,KIBumsBG,gBkBvmsBH,ElBumsBS,KAAM,OkBvmsBf,C;QIBwmsBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,UAAK,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;+EAgBA,yB;MAAA,gE;MkBrnsBA,iB;MIBqnsBA,8C;QAQI,WkBvnsBO,MAAO,KIBunsBG,gBkBvnsBH,ElBunsBS,KAAM,OkBvnsBf,C;QIBwnsBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,UAAK,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;+EAgBA,yB;MAAA,gE;MkBrnsBA,iB;MIBqnsBA,8C;QAQI,WkBvosBO,MAAO,KIBuosBG,gBkBvosBH,ElBuosBS,KAAM,OkBvosBf,C;QIBwosBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,UAAK,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;+EAgBA,yB;MAAA,gE;MkBrnsBA,iB;MIBqnsBA,8C;QAQI,WkBvpsBO,MAAO,KIBupsBG,gBkBvpsBH,ElBupsBS,KAAM,OkBvpsBf,C;QIBwpsBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,UAAK,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;+EAgBA,yB;MAAA,gE;MkBrnsBA,iB;MIBqnsBA,8C;QAQI,WkBvqsBO,MAAO,KIBuqsBG,gBkBvqsBH,ElBuqsBS,KAAM,OkBvqsBf,C;QIBwqsBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,UAAK,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;+EAgBA,yB;MAAA,gE;MkBrnsBA,iB;MIBqnsBA,8C;QAQI,WkBvrsBO,MAAO,KIBursBG,gBkBvrsBH,ElBursBS,KAAM,OkBvrsBf,C;QIBwrsBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,UAAK,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;+EAgBA,yB;MAAA,gE;MkBrnsBA,iB;MIBqnsBA,8C;QAQI,WkBvssBO,MAAO,KIBussBG,gBkBvssBH,ElBussBS,KAAM,OkBvssBf,C;QIBwssBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,UAAK,CAAL,CAAV,EAAMB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;+EAgBA,yB;MAAA,gE;MAAA,oC;MkBrnsBA,iB;MIBqnsBA,8C;QAQI,WkBvtsBO,MAAO,KIButsBG,gBkBvtsBH,ElButsBS,KAAM,OkBvtsBf,C;QIBwtsBd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,sBAAK,CAAL,EAAMB,EAAMB,kBAAM,CAAN,EAAnB,CAAJ,C;;QAET,OAAO,I;O;KAbX,C;IAGBA,4F;MAQ8D,yB;QAAA,YAA0B,I;MAAM,sB;QAAA,SAAuB,E;MAAI,uB;QAAA,UAAwB,E;MAAI,qB;QAAA,QAAa,E;MAAI,yB;QAAA,YAA0B,K;MAAO,yB;QAAA,YAAoC,I;MAGvN,Q;MAFhB,MAAO,gBAAO,MAAP,C;MACP,YAAY,C;MACZ,wBAAgB,

SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,IAAI,iCAAU,CAAd,C;UAAiB,MAAO,gBAAO,SAAP,C;QACxB,IAAI,QAAQ,CAAR,IAAa,SAAS,KAA1B,C;UACW,gBAAP,MAAO,EAAC,OAAd,EAaUB,SAAvB,C;;UACJ,K;;MAEX,IAAI,SAAS,CAAT,IAAc,QAAQ,KAA1B,C;QAAiC,MAAO,gBAAO,SAAP,C;MACxC,MAAO,gBAAO,OAA P,C;MACP,OAAO,M;K;IAGX,8F;MAQwD,yB;QAAA,YAA0B,I;MAAM,sB;QAAA,SAAuB,E;MAAI,uB;QAAA, UAAwB,E;MAAI,qB;QAAA,QAAa,E;MAAI,yB;QAAA,YAA0B,K;MAAO,yB;QAAA,YAAuC,I;MAGpN,Q;MA FhB,MAAO,gBAAO,MAAP,C;MACP,YAAY,C;MACZ,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,I AAI,iCAAU,CAAd,C;UAAiB,MAAO,gBAAO,SAAP,C;QACxB,IAAI,QAAQ,CAAR,IAAa,SAAS,KAA1B,C;UA CI,IAAI,iBAAJ,C;YACI,MAAO,gBAAO,UAAU,OAAV,CAAP,C;;YAEP,MAAO,gBAAO,OAAQ,WAAf,C;;UAC R,K;;MAEX,IAAI,SAAS,CAAT,IAAc,QAAQ,KAA1B,C;QAAiC,MAAO,gBAAO,SAAP,C;MACxC,MAAO,gBAA O,OAA P,C;MACP,OAAO,M;K;IAGX,8F;MAQyD,yB;QAAA,YAA0B,I;MAAM,sB;QAAA,SAAuB,E;MAAI,u B;QAAA,UAAwB,E;MAAI,qB;QAAA,QAAa,E;MAAI,yB;QAAA,YAA0B,K;MAAO,yB;QAAA,YAAwC,I;MAG tN,Q;MAFhB,MAAO,gBAAO,MAAP,C;MACP,YAAY,C;MACZ,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB, M;QACI,IAAI,iCAAU,CAAd,C;UAAiB,MAAO,gBAAO,SAAP,C;QACxB,IAAI,QAAQ,CAAR,IAAa,SAAS,KA A1B,C;UACI,IAAI,iBAAJ,C;YACI,MAAO,gBAAO,UAAU,OAAV,CAAP,C;;YAEP,MAAO,gBAAO,OAAQ,WA Af,C;;UACR,K;;MAEX,IAAI,SAAS,CAAT,IAAc,QAAQ,KAA1B,C;QAAiC,MAAO,gBAAO,SAAP,C;MACxC, MAAO,gBAAO,OAA P,C;MACP,OAAO,M;K;IAGX,8F;MAQuD,yB;QAAA,YAA0B,I;MAAM,sB;QAAA,SAAu B,E;MAAI,uB;QAAA,UAAwB,E;MAAI,qB;QAAA,QAAa,E;MAAI,yB;QAAA,YAA0B,K;MAAO,yB;QAAA,YA AsC,I;MAGIN,Q;MAFhB,MAAO,gBAAO,MAAP,C;MACP,YAAY,C;MACZ,wBAAGB,SAAhB,gB;QAAGB,cAA A,SAAhB,M;QACI,IAAI,iCAAU,CAAd,C;UAAiB,MAAO,gBAAO,SAAP,C;QACxB,IAAI,QAAQ,CAAR,IAAa, SAAS,KAA1B,C;UACI,IAAI,iBAAJ,C;YACI,MAAO,gBAAO,UAAU,OAAV,CAAP,C;;YAEP,MAAO,gBAAO, OAAQ,WAAf,C;;UACR,K;;MAEX,IAAI,SAAS,CAAT,IAAc,QAAQ,KAA1B,C;QAAiC,MAAO,gBAAO,SAAP, C;MACxC,MAAO,gBAAO,OAA P,C;MACP,OAAO,M;K;IAGX,8F;MAQwD,yB;QAAA,YAA0B,I;MAAM,sB;Q AAA,SAAuB,E;MAAI,uB;QAAA,UAAwB,E;MAAI,qB;QAAA,QAAa,E;MAAI,yB;QAAA,YAA0B,K;MAAO,yB;QAAA,YA AuC,I;MAGpN,Q;MAFhB,MAAO,gBAAO,MAAP,C;MACP,YAAY,C;MACZ,wBAAGB,SAAhB,gB; QAAGB,cAAA,SAAhB,M;QACI,IAAI,iCAAU,CAAd,C;UAAiB,MAAO,gBAAO,SAAP,C;QACxB,IAAI,QAAQ, CAAR,IAAa,SAAS,KAA1B,C;UACI,IAAI,iBAAJ,C;YACI,MAAO,gBAAO,UAAU,OAAV,CAAP,C;;YAEP,MA AAO,gBAAO,OAAQ,WAAf,C;;UACR,K;;MAEX,IAAI,SAAS,CAAT,IAAc,QAAQ,KAA1B,C;QAAiC,MAAO,gB AAO,SAAP,C;MACxC,MAAO,gBAAO,OAA P,C;MACP,OAAO,M;K;IAGX,8F;MAQyD,yB;QAAA,YAA0B,I;M AAM,sB;QAAA,SAAuB,E;MAAI,uB;QAAA,UAAwB,E;MAAI,qB;QAAA,QAAa,E;MAAI,yB;QAAA,YAA0B,K ;MAAO,yB;QAAA,YAAwC,I;MAGtN,Q;MAFhB,MAAO,gBAAO,MAAP,C;MACP,YAAY,C;MACZ,wBAAGB, SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,IAAI,iCAAU,CAAd,C;UAAiB,MAAO,gBAAO,SAAP,C;QACxB,IA AI,QAAQ,CAAR,IAAa,SAAS,KAA1B,C;UACI,IAAI,iBAAJ,C;YACI,MAAO,gBAAO,UAAU,OAAV,CAAP,C;; YAEP,MAAO,gBAAO,OAAQ,WAAf,C;;UACR,K;;MAEX,IAAI,SAAS,CAAT,IAAc,QAAQ,KAA1B,C;QAAiC, MAAO,gBAAO,SAAP,C;MACxC,MAAO,gBAAO,OAA P,C;MACP,OAAO,M;K;IAGX,8F;MAQ0D,yB;QAAA,Y AA0B,I;MAAM,sB;QAAA,SAAuB,E;MAAI,uB;QAAA,UAAwB,E;MAAI,qB;QAAA,QAAa,E;MAAI,yB;QAAA, YAA0B,K;MAAO,yB;QAAA,YAAyC,I;MAGxN,Q;MAFhB,MAAO,gBAAO,MAAP,C;MACP,YAAY,C;MACZ, wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,IAAI,iCAAU,CAAd,C;UAAiB,MAAO,gBAAO,SAAP,C; QACxB,IAAI,QAAQ,CAAR,IAAa,SAAS,KAA1B,C;UACI,IAAI,iBAAJ,C;YACI,MAAO,gBAAO,UAAU,OAAV, CAAP,C;;YAEP,MAAO,gBAAO,OAAQ,WAAf,C;;UACR,K;;MAEX,IAAI,SAAS,CAAT,IAAc,QAAQ,KAA1B,C ;QAAiC,MAAO,gBAAO,SAAP,C;MACxC,MAAO,gBAAO,OAA P,C;MACP,OAAO,M;K;IAGX,8F;MAQ2D,yB; QAAA,YAA0B,I;MAAM,sB;QAAA,SAAuB,E;MAAI,uB;QAAA,UAAwB,E;MAAI,qB;QAAA,QAAa,E;MAAI,y B;QAAA,YAA0B,K;MAAO,yB;QAAA,YAA0C,I;MAG1N,Q;MAFhB,MAAO,gBAAO,MAAP,C;MACP,YAAY, C;MACZ,wBAAGB,SAAhB,gB;QAAGB,cAAA,SAAhB,M;QACI,IAAI,iCAAU,CAAd,C;UAAiB,MAAO,gBAAO, SAAP,C;QACxB,IAAI,QAAQ,CAAR,IAAa,SAAS,KAA1B,C;UACI,IAAI,iBAAJ,C;YACI,MAAO,gBAAO,UAA U,OAAV,CAAP,C;;YAEP,MAAO,gBAAO,OAAQ,WAAf,C;;UACR,K;;MAEX,IAAI,SAAS,CAAT,IAAc,QAAQ, KAA1B,C;QAAiC,MAAO,gBAAO,SAAP,C;MACxC,MAAO,gBAAO,OAA P,C;MACP,OAAO,M;K;IAGX,8F;M AQwD,yB;QAAA,YAA0B,I;MAAM,sB;QAAA,SAAuB,E;MAAI,uB;QAAA,UAAwB,E;MAAI,qB;QAAA,QAAa, E;MAAI,yB;QAAA,YAA0B,K;MAAO,yB;QAAA,YAAuC,I;MAGpN,Q;MAFhB,MAAO,gBAAO,MAAP,C;MAC

P, YAAy, C; MACZ, wBAAGb, SAAhb, gB; QAAgB, cAAhb, UAAgB, SAAhb, O; QACI, IAAI, iCAAU, CAAd, C; UAAi B, MAAO, gBAAO, SAAP, C; QACxB, IAAI, QAAQ, CAAR, IAAa, SAAS, KAAIB, C; UACI, IAAI, iBAAJ, C; YACI, MA AO, gBAAO, UAAU, oBAAV, CAAP, C; YAEP, MAAO, gBAAO, OAAP, C; UACR, K; MAEX, IAAI, SAAS, CAAT, IA Ac, QAAQ, KAAIB, C; QAAiC, MAAO, gBAAO, SAAP, C; MACxC, MAAO, gBAAO, OAAP, C; MACP, OAAO, M; K; I AGX, 0F; MAQyC, yB; QAAA, YAA0B, I; MAAM, sB; QAAA, SAAuB, E; MAAI, uB; QAAA, UAAwB, E; MAAI, qB; QA AA, QAAa, E; MAAI, yB; QAAA, YAA0B, K; MAAO, yB; QAAA, YAAoC, I; MACIN, OAAO, kBAAO, sBAAP, EAAwB , SAAxB, EAAmC, MAAnC, EAA2C, OAA3C, EAAoD, KAApD, EAA2D, SAA3D, EAASe, SAAte, CAaiF, W; K; IAG5 F, 4F; MAQkC, yB; QAAA, YAA0B, I; MAAM, sB; QAAA, SAAuB, E; MAAI, uB; QAAA, UAAwB, E; MAAI, qB; QAAA, QAAa, E; MAAI, yB; QAAA, YAA0B, K; MAAO, yB; QAAA, YAAuC, I; MAC9M, OAAO, oBAAO, sBAAP, EAAwB, S AAxB, EAAmC, MAAnC, EAA2C, OAA3C, EAAoD, KAApD, EAA2D, SAA3D, EAASe, SAAte, CAaiF, W; K; IAG5F, 4F; MAQmC, yB; QAAA, YAA0B, I; MAAM, sB; QAAA, SAAuB, E; MAAI, uB; QAAA, UAAwB, E; MAAI, qB; QAAA, QAAa, E; MAAI, yB; QAAA, YAA0B, K; MAAO, yB; QAAA, YAAwC, I; MACHN, OAAO, oBAAO, sBAAP, EAAwB, S AAxB, EAAmC, MAAnC, EAA2C, OAA3C, EAAoD, KAApD, EAA2D, SAA3D, EAASe, SAAte, CAaiF, W; K; IAG5F, 4F; MAQiC, yB; QAAA, YAA0B, I; MAAM, sB; QAAA, SAAuB, E; MAAI, uB; QAAA, UAAwB, E; MAAI, qB; QAAA, Q AAa, E; MAAI, yB; QAAA, YAA0B, K; MAAO, yB; QAAA, YAAcC, I; MAC5M, OAAO, oBAAO, sBAAP, EAAwB, SAA xB, EAAmC, MAAnC, EAA2C, OAA3C, EAAoD, KAApD, EAA2D, SAA3D, EAASe, SAAte, CAaiF, W; K; IAG5F, 4F; MAQkC, yB; QAAA, YAA0B, I; MAAM, sB; QAAA, SAAuB, E; MAAI, uB; QAAA, UAAwB, E; MAAI, qB; QAAA, QAA a, E; MAAI, yB; QAAA, YAA0B, K; MAAO, yB; QAAA, YAAuC, I; MAC9M, OAAO, oBAAO, sBAAP, EAAwB, SAAxB , EAAmC, MAAnC, EAA2C, OAA3C, EAAoD, KAApD, EAA2D, SAA3D, EAASe, SAAte, CAaiF, W; K; IAG5F, 4F; M AQmC, yB; QAAA, YAA0B, I; MAAM, sB; QAAA, SAAuB, E; MAAI, uB; QAAA, UAAwB, E; MAAI, qB; QAAA, QAAa, E; MAAI, yB; QAAA, YAA0B, K; MAAO, yB; QAAA, YAAwC, I; MACHN, OAAO, oBAAO, sBAAP, EAAwB, SAAxB, E AA mC, MAAnC, EAA2C, OAA3C, EAAoD, KAApD, EAA2D, SAA3D, EAASe, SAAte, CAaiF, W; K; IAG5F, 4F; MAQ oC, yB; QAAA, YAA0B, I; MAAM, sB; QAAA, SAAuB, E; MAAI, uB; QAAA, UAAwB, E; MAAI, qB; QAAA, QAAa, E; M AA I, yB; QAAA, YAA0B, K; MAAO, yB; QAAA, YAAyC, I; MACIN, OAAO, oBAAO, sBAAP, EAAwB, SAAxB, EAAm C, MAAnC, EAA2C, OAA3C, EAAoD, KAApD, EAA2D, SAA3D, EAASe, SAAte, CAaiF, W; K; IAG5F, 4F; MAQqC, y B; QAAA, YAA0B, I; MAAM, sB; QAAA, SAAuB, E; MAAI, uB; QAAA, UAAwB, E; MAAI, qB; QAAA, QAAa, E; MAAI , yB; QAAA, YAA0B, K; MAAO, yB; QAAA, YAAoC, I; MACpN, OAAO, oBAAO, sBAAP, EAAwB, SAAxB, EAAmC, MAAnC, EAA2C, OAA3C, EAAoD, KAApD, EAA2D, SAA3D, EAASe, SAAte, CAaiF, W; K; IAG5F, 4F; MAQkC, yB; QAAA, YAA0B, I; MAAM, sB; QAAA, SAAuB, E; MAAI, uB; QAAA, UAAwB, E; MAAI, qB; QAAA, QAAa, E; MAAI, y B; QAAA, YAA0B, K; MAAO, yB; QAAA, YAAuC, I; MAC9M, OAAO, oBAAO, sBAAP, EAAwB, SAAxB, EAAmC, M AA nC, EAA2C, OAA3C, EAAoD, KAApD, EAA2D, SAA3D, EAASe, SAAte, CAaiF, W; K; IAQxE, 4C; MAAA, mB; Q AA E, OAAK, qBAAL, eAAK, C; O; K; IAL3B, +B; MAII, IAlleO, qBAAQ, CAkIef, C; QAAe, OAAO, W; MACtB, kCAA g B, 4BAAhB, C; K; IAQgB, 8C; MAAA, mB; QAAE, OAAK, yBAAL, eAAK, C; O; K; IAL3B, iC; MAII, IAlleO, qBAAQ, C AkIef, C; QAAe, OAAO, W; MACtB, kCAAgB, 8BAAhB, C; K; IAQgB, 8C; MAAA, mB; QAAE, OAAK, 0BAAL, eAAK, C; O; K; IAL3B, iC; MAII, IAlleO, qBAAQ, CAkIef, C; QAAe, OAAO, W; MACtB, kCAAgB, 8BAAhB, C; K; IAQgB, 8C; MAAA, mB; QAAE, OAAK, 0BAAL, eAAK, C; O; K; IAL3B, iC; MAII, IAlleO, qBAAQ, CAkIef, C; QAAe, OAAO, W; MACtB, kCAAgB, 8BAAhB, C; K; IAQgB, 8C; MAAA, mB; QAAE, OAAK, yBAAL, eAAK, C; O; K; IAL3B, iC; MAII, IAlleO, qBAAQ, CAkIef, C; QAAe, OAAO, W; MACtB, kCAAgB, 8BAAh B, C; K; IAUGB, 4C; MAAA, mB; QAAE, OAAK, qBAAL, eAAK, C; O; K; IAP3B, +B; MAMI, IA5peO, qBAAQ, CA4pef, C ; QAAe, OAAO, e; MACtB, kCAAgB, 4BAAhB, C; K; IAUGB, 8C; MAAA, mB; QAAE, OAAK, yBAAL, eAAK, C; O; K; IA P3B, iC; MAMI, IA9peO, qBAAQ, CA8pef, C; QAAe, OAAO, e; MACtB, kCAAgB, 8BAAhB, C; K; IAUGB, 8C; MAAA, m B; QAAE, OAAK, 0BAAL, eAAK, C; O; K; IAP3B, iC; MAMI, IAhqeO, qBAAQ, CAgqef, C; QAAe, OAAO, e; MACtB, kC AA gB, 8BAAhB, C; K; IAUGB, 8C; MAAA, mB; QAAE, OAAK, wBAAL, eAAK, C; O; K; IAP3B, iC; MAMI, IAIqeO, qBA

AQ,CAkqef,C;QAAe,OAAO,e;MACtB,kCAAgB,8BAAhB,C;K;IAUgB,8C;MAAA,mB;QAAE,OAAK,yBAAL,eA  
AK,C;O;K;IAP3B,iC;MAMI,IApqeO,qBAAQ,CAoqef,C;QAAe,OAAO,e;MACtB,kCAAgB,8BAAhB,C;K;IAUgB,  
8C;MAAA,mB;QAAE,OAAK,0BAAL,eAAK,C;O;K;IAP3B,iC;MAMI,IAtqeO,qBAAQ,CAsqef,C;QAAe,OAAO,e  
;MACtB,kCAAgB,8BAAhB,C;K;IAUgB,8C;MAAA,mB;QAAE,OAAK,2BAAL,eAAK,C;O;K;IAP3B,iC;MAMI,I  
AxqeO,qBAAQ,CAwqef,C;QAAe,OAAO,e;MACtB,kCAAgB,8BAAhB,C;K;IAUgB,8C;MAAA,mB;QAAE,OAA  
K,4BAAL,eAAK,C;O;K;IAP3B,iC;MAMI,IA1qeO,qBAAQ,CA0qef,C;QAAe,OAAO,e;MACtB,kCAAgB,8BAAh  
B,C;K;IAUgB,8C;MAAA,mB;QAAE,OAAK,yBAAL,eAAK,C;O;K;IAP3B,iC;MAMI,IA5qeO,qBAAQ,CA4qef,C;  
QAAe,OAAO,e;MACtB,kCAAgB,8BAAhB,C;K;IAGJ,4B;MAOoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MA  
CjB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,OAAO,O;QACP,qB;;MAEJ,OAAW,UAAS,CAAb,G  
AAgB,wCAAO,IAAvB,GAAgC,MAAM,K;K;IAGjD,8B;MAOoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MACj  
B,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,OAAO,O;QACP,qB;;MAEJ,OAAW,UAAS,CAAb,GAA  
gB,wCAAO,IAAvB,GAAgC,MAAM,K;K;IAGjD,8B;MAOoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MACjB,  
wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,OAAO,O;QACP,qB;;MAEJ,OAAW,UAAS,CAAb,GAA  
gB,wCAAO,IAAvB,GAAgC,MAAM,K;K;IAGjD,8B;MAOoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MACjB,w  
BAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,OAAO,O;QACP,qB;;MAEJ,OAAW,UAAS,CAAb,GAAgB,  
wCAAO,IAAvB,GAAgC,MAAM,K;K;IAGjD,8B;MAOoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MACjB,wB  
AAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,OAAO,O;QACP,qB;;MAEJ,OAAW,UAAS,CAAb,GAAgB,w  
CAAO,IAAvB,GAAgC,MAAM,K;K;IAGjD,8B;MAMoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MACjB,wBAA  
gB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,OAAO,O;QACP,qB;;MAEJ,OAAW,UAAS,CAAb,GAAgB,wCA  
AO,IAAvB,GAAgC,MAAM,K;K;IAGjD,8B;MAMoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MACjB,wBAAgB,  
SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,OAAO,O;QACP,qB;;MAEJ,OAAW,UAAS,CAAb,GAAgB,wCAA  
O,IAAvB,GAAgC,MAAM,K;K;IAGjD,8B;MAMoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MACjB,wBAAgB,  
SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,OAAO,O;QACP,qB;;MAEJ,OAAW,UAAS,CAAb,GAAgB,wCAAO,  
IAAvB,GAAgC,MAAM,K;K;IAGjD,8B;MAMoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MACjB,wBAAgB,S  
AAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,OAAO,O;QACP,qB;;MAEJ,OAAW,UAAS,CAAb,GAAgB,wCAAO,I  
AAvB,GAAgC,MAAM,K;K;IAGjD,8B;MAMoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MACjB,wBAAgB,SA  
AhB,gB;QAAgB,cAAA,SAAhB,M;QACI,OAAO,O;QACP,qB;;MAEJ,OAAW,UAAS,CAAb,GAAgB,wCAAO,IA  
AvB,GAAgC,MAAM,K;K;IAGjD,+B;MAMoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MACjB,wBAAgB,SAA  
hB,gB;QAAgB,cAAA,SAAhB,M;QACI,OAAO,O;QACP,qB;;MAEJ,OAAW,UAAS,CAAb,GAAgB,wCAAO,IAA  
vB,GAAgC,MAAM,K;K;IAGjD,wB;MAMoB,Q;MADhB,UAAe,C;MACf,wBAAgB,SAAhB,gB;QAAgB,cAAA,S  
AAhB,M;QACI,YAAO,O;;MAEX,OAAO,G;K;IAGX,0B;MAMoB,Q;MADhB,UAAe,C;MACf,wBAAgB,SAAhB,  
gB;QAAgB,cAAA,SAAhB,M;QACI,YAAO,O;;MAEX,OAAO,G;K;IAGX,0B;MAMoB,Q;MADhB,UAAe,C;MA  
Cf,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,YAAO,OAAP,I;;MAEJ,OAAO,G;K;IAGX,0B;MAMo  
B,Q;MADhB,Y;MACA,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,cAAO,OAAP,C;;MAEJ,OAAO,G  
;K;IAGX,0B;MAMoB,Q;MADhB,UAAiB,G;MACjB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,OA  
AO,O;;MAEX,OAAO,G;K;IAGX,0B;MAMoB,Q;MADhB,UAAkB,G;MACIB,wBAAgB,SAAhB,gB;QAAgB,cA  
AA,SAAhB,M;QACI,OAAO,O;;MAEX,OAAO,G;K;IAGX,0B;MAKoB,Q;MADhB,UAAe,C;MACf,wBAAgB,SA  
AhB,gB;QAAgB,cAAA,SAAhB,M;QACI,YAAO,O;;MAEX,OAAO,G;K;IAGX,0B;MAKoB,Q;MADhB,UAAe,C;  
MACf,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,YAAO,O;;MAEX,OAAO,G;K;IAGX,0B;MAKoB,  
Q;MADhB,UAAe,C;MACf,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,YAAO,OAAP,I;;MAEJ,OAA  
O,G;K;IAGX,0B;MAKoB,Q;MADhB,Y;MACA,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,cAAO,O  
AAP,C;;MAEJ,OAAO,G;K;IAGX,0B;MAKoB,Q;MADhB,UAAiB,G;MACjB,wBAAgB,SAAhB,gB;QAAgB,cAA  
A,SAAhB,M;QACI,OAAO,O;;MAEX,OAAO,G;K;IAGX,2B;MAKoB,Q;MADhB,UAAkB,G;MACIB,wBAAgB,S  
AAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,OAAO,O;;MAEX,OAAO,G;K;Ia5uuBX,oD;MAQuF,wC;K;IARvF,8C  
ASI,Y;MAAuC,8B;K;IAT3C,gF;4FOOA,qB;MAOI,OAAO,sBAAL,CAAJ,C;K;4FAGX,qB;MAOI,OAAO,sBAAL,  
CAAJ,C;K;4FAGX,qB;MAOI,OAAO,sBAAL,CAAJ,C;K;4FAGX,qB;MAOI,OAAO,sBAAL,CAAJ,C;K;4FAGX,q

B;MAOI,OAAO,sBAAI,CAAJ,C;K;IAGX,wC;MAII,IAAI,oCAAJ,C;QACI,OAAO,yBAAS,OAAT,C;MACX,OA  
AO,qBAAQ,OAAR,KAAoB,C;K;IAWG,yC;MAAA,qB;QAAE,MAAM,8BAA0B,iDAA8C,aAA9C,MAA1B,C;O;  
K;IAR1C,qC;MAMI,IAAI,8BAAJ,C;QACI,OAAO,sBAAI,KAAJ,C;MACX,OAAO,6BAAgB,KAAhB,EAAuB,uB  
AAvB,C;K;0FAGX,4B;MAOI,OAAO,sBAAI,KAAJ,C;K;IAGX,2D;MAcqb,Q;MARjB,IAAI,8BAAJ,C;QACI,OA  
AsB,KA4Lf,IAAS,CAAT,IA5Le,KA4LD,IAAS,iBA5LvB,SA4LuB,CAA3B,GA5LI,SA4LkC,aA5LnB,KA4LmB,C  
AAtC,GA5L0B,YA4L4B,CA5LnC,KA4LmC,C;OA3L7D,IAAI,QAAQ,CAAZ,C;QACI,OAAO,aAAa,KAAb,C;M  
ACX,eAAe,oB;MACf,YAA Y,C;MACZ,OAAO,QAAS,UAAhB,C;QACI,cAAc,QAAS,O;QACvB,IAAI,WAAS,Y  
AAT,EAAS,oBAAT,OAAJ,C;UACI,OAAO,O;;MAEf,OAAO,aAAa,KAAb,C;K;sGAGX,yB;MAAA,8D;MAAA,i  
D;QAOI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAsC,sBAAI,KAAJ,CAAtC,GAAsD,aAAa,KAAb,C;O;K  
APjE,C;IAUA,6C;MAcqb,Q;MARjB,IAAI,8BAAJ,C;QACI,OAA Y,YAAL,SAAK,EAAU,KAAV,C;MACHB,IAA  
I,QAAQ,CAAZ,C;QACI,OAAO,I;MACX,eAAe,oB;MACf,YAA Y,C;MACZ,OAAO,QAAS,UAAhB,C;QACI,cAA  
c,QAAS,O;QACvB,IAAI,WAAS,YAAT,EAAS,oBAAT,OAAJ,C;UACI,OAAO,O;;MAEf,OAAO,I;K;sGAGX,yB;  
MAAA,sD;MAAA,mC;QAOI,OAA Y,UAA L,SAAK,EAAU,KAAV,C;O;KAPhB,C;gFAUA,gC;MAOW,sB;;QAu  
HS,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UAAM,IAvHH,SAuHO,CAAU,OAAV,CAAJ,C;YAAw  
B,qBAAO,O;YAAP,uB;;QAC9C,qBAAO,I;;MAxHP,yB;K;wFAGJ,gC;MA2VoB,Q;MADhB,WAAe,I;MACC,2B;  
MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,IARVc,SAqVV,CAAU,OAAV,CAAJ,C;UACI,OAAO,O;;MatVf,  
OAYVO,I;K;wFAtVX,gC;MAOW,qB;;QAwVP,eAAoB,+BAAa,cAAb,C;QACpB,OAAO,QAAS,cAAhB,C;UACI,  
cAAc,QAAS,W;UACvB,IA3Vc,SA2VV,CAAU,OAAV,CAAJ,C;YAAwB,oBAAO,O;YAAP,sB;;QAE5B,oBAAO,  
I;;MA7VP,wB;K;IAGJ,6B;MAMQ,kBADE,SACF,Q;QAAW,OAA Y,SAAL,SAAK,C;;QAE5B,eAAe,oB;QACf,I  
AAI,CAAC,QAAS,UAA d,C;UACI,MAAM,2BAAuB,sBAAvB,C;QACV,OAAO,QAAS,O;;K;IAK5B,6B;MAKI,I  
AAI,mBAAJ,C;QACI,MAAM,2BAAuB,gBAAvB,C;MACV,OAAO,sBAAK,CAAL,C;K;mFAGX,yB;MAAA,iE;  
MAAA,uC;QAKoB,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UAAM,IAAI,UAAU,OAAV,CAAJ,C;Y  
AAwB,OAAO,O;;QACrD,MAAM,gCAAuB,wDAAvB,C;O;KANV,C;oGASA,yB;MAAA,iE;MAAA,uC;QASW,  
Q;QAAA,+B;;UAYS,U;UAAA,6B;UAAhB,OAAGB,gBAAhB,C;YAAgB,2B;YACZ,aAbwB,SAaX,CAAU,OAAV  
,C;YACb,IAAI,cAAJ,C;cACI,8BAAO,M;cAAP,gC;;UAGR,8BAAO,I;;QAIbA,kC;QAAA,iB;UAAmC,MAAM,g  
CAAuB,mEAAvB,C;SAAhD,OAAO,I;O;KATX,C;gHAYA,gC;MASoB,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C  
;QAAGB,yB;QACZ,aAAa,UAAU,OAAV,C;QACb,IAAI,cAAJ,C;UACI,OAAO,M;;MAGf,OAAO,I;K;IAGX,mC;  
MAKQ,kBADE,SACF,Q;QACI,IAAI,mBAAJ,C;UACI,OAAO,I;;UAEP,OAAO,sBAAK,CAAL,C;;QAGX,eAAe,o  
B;QACf,IAAI,CAAC,QAAS,UAA d,C;UACI,OAAO,I;QACX,OAAO,QAAS,O;;K;IAK5B,mC;MAII,OAAW,mBA  
AJ,GAAe,IAAf,GAAYB,sBAAK,CAAL,C;K;-FAGpC,gC;MAIoB,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QA  
AgB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,O;;MACrD,OAAO,I;K;0FAGX,yB;MAAA,8D;M  
AAA,iD;QAKI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAsC,sBAAI,KAAJ,CAAtC,GAAsD,aAAa,KAAb,  
C;O;KALjE,C;IAQA,uC;MAMI,OAAW,SAAS,CAAT,IAAc,SAAS,2BAA3B,GAAsC,sBAAI,KAAJ,CAAtC,GAA  
sD,I;K;IAGjE,uC;MAMiB,Q;MAFb,IAAI,8BAAJ,C;QAakB,OAAO,SAAK,eAAQ,OAAR,C;MAC9B,YAA Y,C;M  
ACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,mBAAmB,KAA nB,C;QACA,IAAI,gBAAW,IAAX,CAAJ,C;UA  
CI,OAAO,K;QACX,qB;;MAEJ,OAAO,E;K;IAGX,uC;MAKI,OAAO,wBAAQ,OAAR,C;K;gGAGX,yB;MAAA,w  
E;MAAA,uC;QAKiB,Q;QADb,YAA Y,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,mBAAmB,KAA nB,  
C;UACA,IAAI,UAAU,IAAV,CAAJ,C;YACI,OAAO,K;UACX,qB;;QAEJ,OAAO,E;O;KAXX,C;gGAcA,gC;MAK  
iB,Q;MADb,YAA Y,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,IAAI,UAAU,IAAV,CAAJ,C;UACI,OA  
AO,K;QACX,qB;;MAEJ,OAAO,E;K;8FAGX,yB;MAAA,wE;MAAA,uC;QAMiB,Q;QAFb,gBAAgB,E;QACHB,Y  
AA Y,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,mBAAmB,KAA nB,C;UACA,IAAI,UAAU,IAAV,CA  
AJ,C;YACI,YAA Y,K;UACHB,qB;;QAEJ,OAAO,S;O;KAZX,C;8FAeA,gC;MAII,eAAe,SAAK,sBAAa,cAAb,C;M  
ACpB,OAAO,QAAS,cAAhB,C;QACI,IAAI,UAAU,QAAS,WAA nB,CAAJ,C;UACI,OAAO,QAAS,Y;;MAGxB,O  
AAO,E;K;IAGX,4B;MASQ,kBADE,SACF,Q;QAAW,OAA Y,QAAL,SAAK,C;;QAE5B,eAAe,oB;QACf,IAAI,CA  
AC,QAAS,UAA d,C;UACI,MAAM,2BAAuB,sBAAvB,C;QACV,WAAW,QAAS,O;QACpB,OAAO,QAAS,UAAh  
B,C;UACI,OAAO,QAAS,O;QACpB,OAAO,I;;K;IAKnB,4B;MAQI,IAAI,mBAAJ,C;QACI,MAAM,2BAAuB,gBA  
AvB,C;MACV,OAAO,sBAAK,2BAAL,C;K;iFAGX,yB;MAAA,iE;MAAA,gB;MAAA,8B;MAAA,uC;QAUoB,U  
AQT,M;QAVP,WAAe,I;QACf,YAA Y,K;QACI,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,IAAI,UAAU,O

AAV,CAAJ,C;YACI,OAAO,O;YACP,QAAQ,I;;QAGhB,IAAI,CAAC,KAAL,C;UAAy,MAAM,gCAAuB,wDAA  
vB,C;QAEIB,OAAO,2E;O;KAlBX,C;iFAqBA,yB;MAAA,iE;MAAA,uC;QAQI,eAAe,SAAK,sBAAa,cAAb,C;QA  
CpB,OAAO,QAAS,cAAhB,C;UACI,cAAc,QAAS,W;UACvB,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;Q  
AEnC,MAAM,gCAAuB,kDAAvB,C;O;KAbV,C;IAGbA,2C;MAOIb,Q;MAHb,IAAI,8BAAJ,C;QAAkB,OAAO,S  
AAK,mBAAY,OAAZ,C;MAC9B,gBAAgB,E;MACHb,YAAy,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QA  
CT,mBAAmB,KAAhB,C;QACA,IAAI,gBAAW,IAAX,CAAJ,C;UACI,YAAy,K;QACHb,qB;;MAEJ,OAAO,S;K;I  
AGX,2C;MAKI,OAAO,4BAAy,OAAZ,C;K;IAGX,kC;MAOQ,kBADE,SACF,Q;QAAW,OAAW,mBAAJ,GAAe,I  
AAf,GAAyB,sBAAK,iBAAO,CAAP,IAAL,C;;QAEvC,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UACI,OAA  
O,I;QACX,WAAW,QAAS,O;QACpB,OAAO,QAAS,UAAhB,C;UACI,OAAO,QAAS,O;QACpB,OAAO,I;;K;IAK  
nB,kC;MAMI,OAAW,mBAAJ,GAAe,IAAf,GAAyB,sBAAK,iBAAO,CAAP,IAAL,C;K;6FAGpC,gC;MAOoB,Q;  
MADhB,WAAe,I;MACC,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,IAAI,UAAU,OAAV,CAAJ,C;UACI,  
OAAO,O;;MAGf,OAAO,I;K;6FAGX,gC;MAMI,eAAe,SAAK,sBAAa,cAAb,C;MACpB,OAAO,QAAS,cAAhB,C;  
QACI,cAAc,QAAS,W;QACvB,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,O;;MAEnC,OAAO,I;K;qFAGX,yB  
;MAAA,mC;MAAA,gD;MAAA,4B;QAQI,OAAO,kBAAO,cAAP,C;O;KARX,C;IAWA,sC;MAOI,IAAI,mBAAJ,C  
;QACI,MAAM,2BAAuB,sBAAvB,C;MACV,OAAO,qBAAU,MAAO,iBAAQ,cAAR,CAAjB,C;K;iGAGX,yB;MA  
AA,mC;MAAA,4D;MAAA,4B;QAOI,OAAO,wBAAa,cAAb,C;O;KAPX,C;IAUA,4C;MAMI,IAAI,mBAAJ,C;QA  
CI,OAAO,I;MACX,OAAO,qBAAU,MAAO,iBAAQ,cAAR,CAAjB,C;K;IAGX,8B;MAKQ,kBADE,SACF,Q;QAA  
W,OAAy,UAAAL,SAAK,C;;QAEhB,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UACI,MAAM,2BAAuB,sBAA  
vB,C;QACV,aAAa,QAAS,O;QACtB,IAAI,QAAS,UAAb,C;UACI,MAAM,gCAAyB,uCAAzB,C;QACV,OAAO,M  
;;K;IAKnB,8B;MAIiB,IAAN,I;MAAA,QAAM,cAAN,C;aACH,C;UAAK,MAAM,2BAAuB,gBAAvB,C;aACX,C;  
UAAK,6BAAK,CAAL,C;UAAAL,K;gBACQ,MAAM,gCAAyB,iCAAzB,C;;MAHIB,W;K;qFAOJ,yB;MAAA,kF;M  
AAA,iE;MAAA,gB;MAAA,8B;MAAA,uC;QAMoB,UAST,M;QAXP,aAAiB,I;QACjB,YAAy,K;QACI,2B;QAAh  
B,OAAgB,cAAhB,C;UAAgB,yB;UACZ,IAAI,UAAU,OAAV,CAAJ,C;YACI,IAAI,KAJ,C;cAAW,MAAM,8BA  
AyB,qDAAzB,C;YACjB,SAAS,O;YACT,QAAQ,I;;QAGhB,IAAI,CAAC,KAAL,C;UAAy,MAAM,gCAAuB,wD  
AAvB,C;QAEIB,OAAO,6E;O;KAFx,C;IAkBA,oC;MAKQ,kBADE,SACF,Q;QAAW,OAAW,mBAAQ,CAAZ,GA  
Ae,sBAAK,CAAL,CAAf,GAA4B,I;;QAEIc,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UACI,OAAO,I;QACX  
,aAAa,QAAS,O;QACtB,IAAI,QAAS,UAAb,C;UACI,OAAO,I;QACX,OAAO,M;;K;IAKnB,oC;MAIL,OAAW,mB  
AAQ,CAAZ,GAAe,sBAAK,CAAL,CAAf,GAA4B,I;K;iGAGvC,gC;MAMoB,Q;MAFhB,aAAiB,I;MACjB,YAAy,  
K;MACI,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,IAAI,UAAU,OAAV,CAAJ,C;UACI,IAAI,KAJ,C;Y  
AAW,OAAO,I;UACIB,SAAS,O;UACT,QAAQ,I;;MAGhB,IAAI,CAAC,KAAL,C;QAAy,OAAO,I;MACnB,OAA  
O,M;K;IAGX,8B;MAoBsC,UAGT,MAHS,EAarB,M;MN7pBb,IAAI,EMooBI,KAAK,CNpoBT,CAAJ,C;QACI,cM  
moBc,sD;QNloBd,MAAM,gCAAyB,OAAQ,WAAjC,C;OMmoBV,IAAI,MAAK,CAAT,C;QAAy,OAAO,mB;MA  
CnB,Q;MACA,IAAI,oCAAJ,C;QACI,iBAAiB,iBAAO,CAAP,I;QACjB,IAAI,cAAc,CAAIB,C;UACI,OAAO,W;Q  
ACX,IAAI,eAAc,CAAIB,C;UACI,OAAO,OAAO,kBAAP,C;QACX,OAAO,iBAAa,UAAb,C;QACP,IAAI,8BAAJ,  
C;UACI,IAAI,sCAAJ,C;YAC0B,qB;YAAtB,iBAAc,CAAd,wB;cACI,IAAK,WAAI,sBAAK,KAAL,CAAJ,C;;YAE  
I,wCAAa,CAAb,C;YAAb,OAAa,gBAAb,C;cAAa,wB;cACT,IAAK,WAAI,IAAJ,C;;UAEB,OAAO,I;;QAIX,OAA  
O,gB;;MAEX,YAAy,C;MACC,6B;MAAb,OAAa,gBAAb,C;QAAa,0B;QACT,IAAI,SAAS,CAAb,C;UAAgB,IAA  
K,WAAI,MAAJ,C;;UAAe,qB;;MAEXc,OAAy,qBAAL,IAAK,C;K;IAGhB,kC;MNnqBI,IAAI,EM2qBI,KAAK,C  
N3qBT,CAAJ,C;QACI,cM0qBc,sD;QNzqBd,MAAM,gCAAyB,OAAQ,WAAjC,C;OM0qBV,OAAO,kBAAgB,gB  
AAV,iBAAO,CAAP,IAAU,EAAC,CAAd,CAAhB,C;K;kGAGX,yB;MAAA,4C;MAAA,qD;MAAA,uC;QAMI,IAA  
I,CAAC,mBAAL,C;UACI,eAAe,+BAAa,cAAb,C;UACf,OAAO,QAAS,cAAhB,C;YACI,IAAI,CAAC,UAAU,QA  
AS,WAAhB,CAAL,C;cACI,OAAO,gBAAK,QAAS,YAAT,GAAuB,CAAvB,IAAL,C;;SAInB,OAAO,W;O;KAdX,  
C;0FAiBA,yB;MAAA,+D;MAAA,uC;QAQiB,Q;QAFb,eAAe,K;QACf,WAAW,gB;QACE,2B;QAAb,OAAa,cAAb  
,C;UAAa,sB;UACT,IAAI,QA AJ,C;YACI,IAAK,WAAI,IAAJ,C;eACJ,IAAI,CAAC,UAAU,IAAV,CAAL,C;YACD  
,IAAK,WAAI,IAAJ,C;YACL,WAAW,I;;QAEhB,OAAO,I;O;KAFx,C;oFAkBA,yB;MAAA,+D;MAAA,uC;QAM  
W,kBAAS,gB;QA2FA,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UAAM,IA3FU,SA2FN,CAAU,OAA  
V,CAAJ,C;YAAwB,WAAy,WAAI,OAAJ,C;;QA3FID,OA4FO,W;O;KAIGX,C;kGASA,yB;MAAA,+D;MA6jCA,  
wE;MA7jCA,uC;QAQW,kBAAgB,gB;QA4jCV,gB;QADb,YAAy,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;

UAhjCT,IAZmC,SAY/B,CAgjCkB,oBAAmB,cAAnB,EAAMb,sBAAnB,UAhjCIB,EAjC+C,IAhJ/C,CAAJ,C;Y  
AA2C,sBAgjCQ,IAhJCR,C;;QAZ/C,OAcO,W;O;KATBX,C;sGAWA,yB;MAkjCA,wE;MALjCA,oD;QAYjCiB,gB;Q  
ADb,YAAy,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UAhjCT,IAAI,UAgjCkB,oBAAmB,cAAnB,EAAMb,s  
BAAnB,UAhjCIB,EAjC+C,IAhJ/C,CAAJ,C;YAA2C,sBAgjCQ,IAhJCR,C;;QAE/C,OAAO,W;O;KAXX,C;wGA  
cA,yB;MAAA,+D;MAAA,sC;QAMW,kBAAMb,gB;QASV,Q;QAAA,2B;QAaHb,OAAgB,cAAhB,C;UAAgB,yB;  
UAAM,IAAI,YAAJ,C;YAAkB,WAAy,WAAI,OAAJ,C;;QATpD,OAuO,W;O;KAhBX,C;4GASA,4C;MAMoB,Q;  
MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,YAAJ,C;UAAkB,WAAy,WAAI,OAAJ,C;;MA  
CpD,OAAO,W;K;0FAGX,yB;MAAA,+D;MAAA,uC;QAMW,kBAAY,gB;QA4BH,Q;QAAA,2B;QAaHb,OAAgB  
,cAAhB,C;UAAgB,yB;UAAM,IAAI,CA5BS,SA4BR,CAAU,OAAV,CAAL,C;YAAyB,WAAy,WAAI,OAAJ,C;;Q  
A5B3D,OA6BO,W;O;KANCX,C;IASA,oC;MAMI,OAAO,6BAAGb,gBAAhB,C;K;IAGX,mD;MAMoB,Q;MAAA,  
2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,eAAJ,C;UAAqB,WAAy,WAAI,OAAJ,C;;MACvD,OA  
AO,W;K;8FAGX,6C;MAMoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,CAAC,UAAU  
,OAAV,CAAL,C;UAAyB,WAAy,WAAI,OAAJ,C;;MAC3D,OAAO,W;K;wFAGX,6C;MAMoB,Q;MAAA,2B;MA  
AhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,WAAy,WAAI,OAAJ,C;;MA  
C1D,OAAO,W;K;IAGX,sC;MAII,IAAI,OAAQ,UAAZ,C;QAAuB,Od3wBe,W;Oc4wBtC,OAA6D,SAAtD,SAAK,i  
BAAQ,OAAQ,MAAhB,EAAuB,OAAQ,aAAR,GAAuB,CAAvB,IAAvB,CAAI,D,C;K;IAGjE,sC;MAOkB,Q;MAH  
d,WAAmB,wBAAR,OAAQ,EAAwB,EAAxB,C;MACnB,IAAI,SAAQ,CAAZ,C;QAAe,OAAO,W;MACtB,WAA  
W,iBAAa,IAAb,C;MACG,yB;MAAd,OAAc,cAAAd,C;QAAc,uB;QACV,IAAK,WAAI,sBAAI,KAJ,CAAJ,C;;MA  
ET,OAAO,I;K;IAGX,8B;MAgBiB,Q;MN51Bb,IAAI,EMo1BI,KAAC,CNp1BT,CAAJ,C;QACI,cMm1Bc,sD;QN11  
Bd,MAAM,gCAAyB,OAAQ,WAAjC,C;OMm1BV,IAAI,MAAK,CAAT,C;QAAy,OAAO,W;MACnB,IAAI,oCA  
AJ,C;QACI,IAAI,KAAC,cAAT,C;UAAe,OAAO,mB;QACtB,IAAI,MAAK,CAAT,C;UAAy,OAAO,OAAO,mBA  
AP,C;OAEvB,YAAy,C;MACZ,WAAW,iBAAa,CAAb,C;MACE,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,IAA  
K,WAAI,IAAJ,C;QACL,IAAI,mCAAW,CAAf,C;UACI,K;;MAER,OAAy,qBAAL,IAAK,C;K;IAGhB,kC;MAeqC  
,IAGhB,I;MNt3BjB,IAAI,EM42BI,KAAC,CN52BT,CAAJ,C;QACI,cM22Bc,sD;QN12Bd,MAAM,gCAAyB,OAA  
Q,WAAjC,C;OM22BV,IAAI,MAAK,CAAT,C;QAAy,OAAO,W;MACnB,WAAW,c;MACX,IAAI,KAAC,IAAT,  
C;QAAe,OAAO,mB;MACtB,IAAI,MAAK,CAAT,C;QAAy,OAAO,OAAO,kBAAP,C;MACnB,WAAW,iBAAa,C  
AAb,C;MACX,IAAI,sCAAJ,C;QACI,iBAAc,OAAO,CAAP,IAAd,UAA6B,IAA7B,U;UACI,IAAK,WAAI,sBAAK  
,KAAL,CAAJ,C;;QAEI,sCAAa,OAAO,CAAP,IAAb,C;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,IAAK,WAAI,IAA  
J,C;;MAEb,OAAO,I;K;kGAGX,yB;MAAA,qD;MAAA,gE;MAAA,gD;MAAA,uC;QAMI,IAAI,mBAAJ,C;UACI,  
OAAO,W;QACX,eAAe,+BAAa,cAAb,C;QACf,OAAO,QAAS,cAAhB,C;UACI,IAAI,CAAC,UAAU,QAAS,WAA  
nB,CAAL,C;YACI,QAAS,O;YACT,mBAAMb,iBAAO,QAAS,YAAhB,I;YACnB,IAAI,iBAAGb,CAApB,C;cAAu  
B,OAAO,W;YACI,kBAA3B,eAAa,YAAb,C;YACH,OAAgB,kBAAhB,C;cACI,sBAaA,eAAb,C;YAFR,OH11BD,  
W;;QGg2BP,OAAO,iB;O;KApBX,C;0FAuBA,yB;MAAA,+D;MAAA,uC;QAOiB,Q;QADb,WAAW,gB;QACE,2B  
;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,IAAI,CAAC,UAAU,IAAV,CAAL,C;YACI,K;UACJ,IAAK,WAAI,IAAJ,  
C;;QAET,OAAO,I;O;KAZX,C;IAoBA,+B;MAII,IAAI,wCAAsB,kBAAQ,CAAIC,C;QAAqC,OAAO,mB;MAC5C,  
WAAW,0B;MACN,WAAI,IAAK,C;MACL,OAAO,I;K;IAGX,uC;MAOI,aAAU,2BAAV,OAA2B,CAA3B,M;QA  
CI,QAAQ,MAAO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,sBAAK,CAAL,EAAU,SAAK,aAAI,CAAJ,EAAO,sBAAK,  
CAAL,CAAP,CAAf,C;;K;oFAIR,yB;MAAA,oD;MJn4BA,sC;MAAA,oC;MAAA,uBAOe,yB;QArEf,8D;eAqEe,4B  
;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA5Dd,cAAc,SA4DgB,CA5DhB,CAAd,EAA2B,SA4DM,CA5DN,CAA3B  
,C;W;S;OA4DI,C;MI43Bf,sC;QAMI,IAAI,iBAAO,CAAX,C;UAAc,oBJl4Bd,eAAW,iBIk4BsB,QJl4BtB,CAAX,CI  
k4Bc,C;U;KANIB,C;wGASA,yB;MAAA,oD;MJz3BA,sC;MAAA,oC;MAAA,iCAOe,yB;QAxFf,8D;eAwFe,4B;U  
AAA,uB;YAAU,eAAsB,gB;YAAtB,OA/Ed,cAAc,SA+EgB,CA/EhB,CAAd,EAA2B,SA+EM,CA/EN,CAA3B,C;  
W;S;OA+EI,C;MIk3Bf,sC;QAMI,IAAI,iBAAO,CAAX,C;UAAc,oBJx3Bd,eAAW,2BIw3BgC,QJx3BhC,CAAX,CI  
w3Bc,C;U;KANIB,C;IASA,sC;MAMI,sBAAS,cAAT,C;K;IAGJ,6B;MASgB,Q;MAHZ,IAAI,oCAAJ,C;QACI,IAA  
I,kBAAQ,CAAZ,C;UAAe,OAAy,SAAL,SAAK,C;QAEwB,kBAA3C,sBC5+Bsd,sBD4+BtD,uB;QAAMd,mB;QA  
A3D,OAAoE,OH17BjE,WGk7BiE,C;OAEjD,kBAAhB,0B;MAAwB,oB;MAA/B,OHp7BO,W;K;wFGu7BX,yB;M  
AAA,wD;MJ56BA,sC;MAAA,oC;MAAA,uBAOe,yB;QArEf,8D;eAqEe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB  
,OA5Dd,cAAc,SA4DgB,CA5DhB,CAAd,EAA2B,SA4DM,CA5DN,CAA3B,C;W;S;OA4DI,C;MIq6Bf,sC;QAQI,O

AAO,sBJ76BP,eAAW,iBI66BiB,QJ76BjB,CAAX,CI66BO,C;O;KARX,C;4GAWA,yB;MAAA,wD;MJp6BA,sC;M  
AAA,oC;MAAA,iCAOe,yB;QAxFf,8D;eAwFe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA/Ed,cAAc,SA+EgB,C  
A/EhB,CAAd,EAA2B,SA+EM,CA/EN,CAA3B,C;W;S;OA+EI,C;MI65Bf,sC;QAMI,OAAO,sBjN6BP,eAAW,2BI  
m6B2B,QJn6B3B,CAAX,CI66BO,C;O;KANX,C;IASA,uC;MAMI,OAAO,wBAAW,cAAX,C;K;IAGX,6C;MASE,  
Q;MAHX,IAAI,oCAAJ,C;QACG,IAAI,kBAAQ,CAAZ,C;UAAe,OAAy,SAAL,SAAK,C;QAEe,kBAAIC,sBCvhC  
uD,sBDuhCvD,uB;QAA0C,iC;QAAID,OAAyE,OH79BrE,WG69BqE,C;OAErD,kBAAhB,0B;MAAwB,mC;MAA/  
B,OH/9BO,W;K;IGk+BX,qC;MAMoB,UACL,M;MAHX,aAAa,oBAAa,cAAb,C;MACb,YAAy,C;MACI,2B;MAA  
hB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,OAAO,cAAP,EAAO,sBAAP,YAAkB,O;;MACtB,OAAO,M;K;IAGX,k  
C;MAMoB,UACL,M;MAHX,aAAa,cAAU,cAAV,C;MACb,YAAy,C;MACI,2B;MAAhB,OAAgB,cAAhB,C;QAA  
gB,yB;QACZ,OAAO,cAAP,EAAO,sBAAP,YAAkB,O;;MACtB,OAAO,M;K;IAGX,kC;MAMoB,UACL,M;MAH  
X,aAAa,iBAAU,cAAV,C;MACb,YAAy,C;MACI,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,oC;QACZ,OAAO,cAA  
P,EAAO,sBAAP,YAAkB,O;;MACtB,OAAO,M;K;IAGX,oC;MAMoB,UACL,M;MAHX,aAAa,iBAAy,cAAZ,C;  
MACb,YAAy,C;MACI,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,OAAO,cAAP,EAAO,sBAAP,YAAkB,  
O;;MACtB,OAAO,M;K;IAGX,mC;MAMoB,UACL,M;MAHX,aAAa,iBAAW,cAAX,C;MACb,YAAy,C;MACI,2  
B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,OAAO,cAAP,EAAO,sBAAP,YAAkB,O;;MACtB,OAAO,M;K;I  
AGX,iC;MAMoB,UACL,M;MAHX,aAAa,eAAS,cAAT,C;MACb,YAAy,C;MACI,2B;MAAhB,OAAgB,cAAhB,C  
;QAAgB,yB;QACZ,OAAO,cAAP,EAAO,sBAAP,YAAkB,O;;MACtB,OAAO,M;K;IAGX,kC;MAMoB,UACL,M;  
MAHX,aAAa,iBAAU,cAAV,C;MACb,YAAy,C;MACI,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,OAA  
O,cAAP,EAAO,sBAAP,YAAkB,O;;MACtB,OAAO,M;K;IAGX,mC;MAMoB,UACL,M;MAHX,aAAa,eAAW,cA  
AX,C;MACb,YAAy,C;MACI,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,OAAO,cAAP,EAAO,sBAAP,Y  
AAkB,O;;MACtB,OAAO,M;K;0FAGX,yB;MAAA,kF;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,uC;QAWI,eAA  
wD,cAAzC,YAAy,mCAAwB,EAAxB,CAAZ,CAAYC,EAAc,EAAd,C;QACjD,kBAAY,mBAAoB,QAApB,C;QA  
yEH,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WA1E8C,SA0E/B,CAAU,OAAV,C;UbpkBnB,  
wBAAI,IAAK,MAAT,EAAgB,IAAK,OAARb,C;;Qa0fA,OA4EO,W;O;KAXFX,C;+FAeA,yB;MAAA,kF;MAAA,0  
D;MAAA,yD;MAAA,uE;MAAA,yC;QAWI,eAAwD,cAAzC,YAAy,mCAAwB,EAAxB,CAAZ,CAAYC,EAAc,EA  
Ad,C;QACjD,kBAAc,mBAAoB,QAApB,C;QA2BL,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,  
WAAY,aA5BoC,WA4BhC,CAAY,OAAZ,CAAJ,EAA0B,OAA1B,C;;QA5BhB,OA8BO,W;O;KA1CX,C;+FAeA,y  
B;MAAA,kF;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,yD;QAUl,eAAwD,cAAzC,YAAy,mCAAwB,EAAxB,C  
AAZ,CAAYC,EAAc,EAAd,C;QACjD,kBAAc,mBAAoB,QAApB,C;QA6BL,Q;QAAA,2B;QAAhB,OAAgB,cAAh  
B,C;UAAgB,yB;UACZ,WAAY,aA9BoC,WA8BhC,CAAY,OAAZ,CAAJ,EA9BiD,cA8BvB,CAAE,OAAf,CAA1B,  
C;;QA9BhB,OA9CO,W;O;KA3CX,C;mGAcA,+C;MAUoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB  
;QACZ,WAAY,aAAI,YAAy,OAAZ,CAAJ,EAA0B,OAA1B,C;;MAEhB,OAAO,W;K;mGAGX,+D;MAUoB,Q;M  
AAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,WAAY,aAAI,YAAy,OAAZ,CAAJ,EAA0B,eAAe,OAAf  
,CAA1B,C;;MAEhB,OAAO,W;K;8FAGX,6C;MASoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QA  
CZ,WAAY,eAAU,OAAV,C;QbpkBnB,wBAAI,IAAK,MAAT,EAAgB,IAAK,OAARb,C;;MaskBA,OAAO,W;K;kG  
AGX,yB;MAAA,kF;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,2C;QAYl,aAAa,mBAA6D,cAAzC,YAAy,mCAA  
wB,EAAxB,CAAZ,CAAYC,EAAc,EAAd,CAA7D,C;QAcG,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;  
UAbO,MAcP,aAAI,OAAJ,EAde,aAcF,CAAc,OAAd,CAAb,C;;QAdhB,OAAuB,M;O;KAb3B,C;sGAgBA,iD;MA  
UoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,WAAY,aAAI,OAAJ,EAAa,cAAc,OAAd,CAA  
b,C;;MAEhB,OAAO,W;K;IAGX,gD;MAIiB,Q;MAAA,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,WAAY,WAA  
I,IAAJ,C;;MAEhB,OAAO,W;K;IAGX,gC;MAII,OAAO,0BAAa,eAAW,YAAy,mCAAwB,EAAxB,CAAZ,CAAX,  
CAAb,C;K;IAGX,6B;MAKqB,IAAN,I;MADX,IAAI,oCAAJ,C;QACW,QAAM,cAAN,C;eACH,C;YAAK,kB;YA  
AL,K;eACA,C;YAAK,cAAW,8BAAJ,GAakB,sBAAI,CAAJ,CAAIB,GAA8B,oBAAW,OAAhD,C;YAAL,K;kBA  
Ca,uBAAL,SAAK,C;YAHV,K;;QAAP,W;OAMJ,OAA4B,qBAAhB,gBAAL,SAAK,CAAgB,C;K;IAGhC,oC;MAII  
,IAAI,oCAAJ,C;QACI,OAAy,gBAAL,SAAK,C;MACHB,OAAO,0BAAa,gBAAb,C;K;IAGX,oC;MAII,OAAO,iB  
AAU,SAAV,C;K;IAGX,4B;MAOqB,IAAN,I;MADX,IAAI,oCAAJ,C;QACW,QAAM,cAAN,C;eACH,C;YAAK,iB  
;YAAL,K;eACA,C;YAAK,aAAU,8BAAJ,GAakB,sBAAK,CAAL,CAAIB,GAA+B,oBAAW,OAAhD,C;YAAL,K;  
kBACQ,iCAAa,qBAAiB,YAAy,cAAZ,CAAjB,CAAb,C;YAHL,K;;QAAP,W;OAMJ,OAAwC,oBAAjC,0BAAa,s

BAAb,CAAiC,C;K;sFAG5C,yB;MAAA,+D;MAwFA,gD;MAxFA,uC;QAMW,kBAAU,gB;QAsFD,Q;QAAA,2B; QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAvF6B,SAuFIB,CAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO, IAAP,C;;QAxFhB,OA0FO,W;O;KAhGX,C;uFASA,yB;MAAA,+D;MA0FA,gD;MA1FA,uC;QAUW,kBAAU,gB; QAwFD,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAzF6B,SAyFIB,CAAU,OAAV,C;UACC, OAAZ,WAAY,EAAO,IAAP,C;;QA1FhB,OA4FO,W;O;KATGX,C;oGAaA,yB;MAAA,+D;MA8BA,wE;MAAA,gD ;MA9BA,uC;QAYW,kBAAiB,gB;QA6BR,gB;QADhB,YAAY,C;QACI,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,y B;UACZ,WA9BoC,SA8BzB,CAAU,oBAAmB,cAAnB,EAAMb,sBAAnB,UAAV,EAAuC,OAAvC,C;UACC,OAA Z,WAAY,EAAO,IAAP,C;;QA/BhB,OAIcO,W;O;KA7CX,C;oGAeA,yB;MAAA,+D;MAiCA,wE;MAAA,gD;MAj CA,uC;QAYW,kBAAiB,gB;QAgCR,gB;QADhB,YAAY,C;QACI,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UA CZ,WAjCoC,SAiCzB,CAAU,oBAAmB,cAAnB,EAAMb,sBAAnB,UAAV,EAAuC,OAAvC,C;UACC,OAAZ,WA AY,EAAO,IAAP,C;;QAIChB,OAoCO,W;O;KAhDX,C;wGAeA,yB;MAAA,wE;MAAA,gD;MAAA,oD;QAWoB, UAC4B,M;QAF5C,YAAY,C;QACI,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAW,UAAU,oBAAmB, cAAnB,EAAMb,sBAAnB,UAAV,EAAuC,OAAvC,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W; O;KafX,C;yGAKBA,yB;MAAA,wE;MAAA,gD;MAAA,oD;QAWoB,UAC4B,M;QAF5C,YAAY,C;QACI,2B;QA AhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAW,UAAU,oBAAmB,cAAnB,EAAMb,sBAAnB,UAAV,EAAuC, OAAvC,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KafX,C;0FAkBA,yB;MAAA,gD;MAAA, oD;QAIoB,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAW,UAAU,OAAV,C;UACC,OAAZ, WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KARX,C;2FAWA,yB;MAAA,gD;MAAA,oD;QAQoB,Q;QAAA,2B ;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAW,UAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;; QAEhB,OAAO,W;O;KAZX,C;uFAeA,yB;MAAA,wE;MAyBA,+D;MAzBA,yC;QASW,kBAAU,oB;QAYBD,Q;Q AAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,UA1BiD,WA0BvC,CAAY,OAAZ,C;UbnCP,U;UADP, YaynCe,WbznCH,WaynCwB,GbznCxB,C;UACL,IAAI,aAAJ,C;YACH,aaunCuC,gB;YAA5B,WbtnCX,aasnCgC,G btnChC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UamnCA,iB;UACA,IAAK,WAAI,OAAJ,C;;QA5BT,OA8BO,W;O;K AvCX,C;uFAYA,yB;MAAA,wE;MA8BA,+D;MA9BA,yD;QAUW,kBAAU,oB;QA8BD,Q;QAAA,2B;QAAhB,OA AgB,cAAhB,C;UAAgB,yB;UACZ,UA/BiD,WA+BvC,CAAY,OAAZ,C;UbnCP,U;UADP,Ya2oCe,Wb3oCH,Wa2 oCwB,Gb3oCxB,C;UACL,IAAI,aAAJ,C;YACH,aaYoCuC,gB;YAA5B,WbxoCX,aawoCgC,GbxoChC,EAAS,MAA T,C;YACA,e;;YAEA,c;;UaqoCA,iB;UACA,IAAK,WAjCyD,cAiCrD,CAAe,OAAf,CAAJ,C;;QAJCT,OAmCO,W; O;KA7CX,C;0FAaA,yB;MAAA,+D;MAAA,sD;QASoB,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UA CZ,UAAU,YAAY,OAAZ,C;UbnCP,U;UADP,YaynCe,WbznCH,WaynCwB,GbznCxB,C;UACL,IAAI,aAAJ,C;Y ACH,aaunCuC,gB;YAA5B,WbtnCX,aasnCgC,GbznChC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UamnCA,iB;UACA, IAAK,WAAI,OAAJ,C;;QAET,OAAO,W;O;KAdX,C;2FAiBA,yB;MAAA,+D;MAAA,sE;QAUoB,Q;QAAA,2B;Q AAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,UAAU,YAAY,OAAZ,C;UbnCP,U;UADP,Ya2oCe,Wb3oCH,Wa2o CwB,Gb3oCxB,C;UACL,IAAI,aAAJ,C;YACH,aaYoCuC,gB;YAA5B,WbxoCX,aawoCgC,GbxoChC,EAAS,MAA T,C;YACA,e;;YAEA,c;;UaqoCA,iB;UACA,IAAK,WAAI,eAAe,OAAf,CAAJ,C;;QAET,OAAO,W;O;KafX,C;4FA kBA,yB;MAAA,kC;MAAA,4C;MAAA,wE;QAQW,sC;QAAA,8C;O;MARX,oDASQ,Y;QAA6C,OAAA,oBAAgB, W;O;MATrE,iDAUQ,mB;QAAoC,gCAAY,OAAZ,C;O;MAV5C,gF;MAAA,yC;QAQI,2D;O;KARJ,C;8EAeA,yB; MAAA,kF;MAAA,gE;MAAA,uC;QAOW,kBAAM,eAAa,mCAAwB,EAAXB,CAAb,C;QAuEA,Q;QAAA,2B;QA Ab,OAAa,cAAb,C;UAAa,sB;UACT,WAAY,WAXEwC,SAwEpC,CAAU,IAAV,CAAJ,C;;QAXehB,OAYEO,W;O; KAhFX,C;4FAUA,yB;MAAA,kF;MAAA,gE;MA+BA,wE;MA/BA,uC;QAOW,kBAAa,eAAa,mCAAwB,EAAXB, CAAb,C;QAgCP,gB;QADb,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,WAAY,WAjC+C,SAiC 3C,CAAU,oBAAmB,cAAnB,EAAMb,sBAAnB,UAAV,EAAuC,IAAvC,CAAJ,C;;QAJChB,OAkCO,W;O;KAZCX, C;0GAUA,yB;MAAA,+D;MAoSA,wE;MApSA,uC;QAOW,kBAAoB,gB;QAoSd,gB;QADb,YAAY,C;QACC,2B; QAAb,OAAa,cAAb,C;UAAa,sB;UA1RSB,U;UAAA,cAVQ,SAUR,CAORT,oBAAmB,cAAnB,EAAMb,sBAAnB, UA1RS,EA0RoB,IA1RpB,W;YAA6C,6B;;QAVhF,OAwo,W;O;KAlBX,C;8GAUA,yB;MA0RA,wE;MA1RA,oD; QAIiB,gB;QADb,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UA1RSB,U;UAAA,wBAORT,oBAAmB, cAAnB,EAAMb,sBAAnB,UA1RS,EA0RoB,IA1RpB,W;YAA6C,6B;;QACHF,OAAO,W;O;KARX,C;+FAWA,yB; MAAA,wE;MAAA,oD;QAQiB,UACoC,M;QAFjD,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT, WAAY,WAAI,UAAU,oBAAmB,cAAnB,EAAMb,sBAAnB,UAAV,EAAuC,IAAvC,CAAJ,C;;QACHB,OAAO,W;

O;KAVX,C;4FAaA,yB;MAAA,+D;MAAA,uC;QAOW,kBAaA,gB;QAwPJ,Q;QAAA,2B;QAAhB,OAAgB,cAAhB ,C;UAAgB,yB;UAhPK,U;UAAA,cARe,SAQf,CAgPQ,OAhPR,W;YAAc,6B;;QAR3D,OASO,W;O;KAhBX,C;gG AUA,yB;MAAA,oD;QAqPoB,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UAhPK,U;UAAA,wBAGPQ, OAhPR,W;YAAc,6B;;QAC3D,OAAO,W;O;KANX,C;kFASA,6C;MAKiB,Q;MAAA,2B;MAAb,OAAa,cAAb,C; QAAa,sB;QACT,WAAy,WAAI,UAAU,IAAV,CAAJ,C;;MACHb,OAAO,W;K;IAQiB,4C;MAAA,mB;QAAE,gC; O;K;IAL9B,gC;MAKI,OAAO,qBAAiB,6BAAjB,C;K;IAGX,+B;MASI,OAA2B,SAAf,eAAL,SAAK,CAAE,C;K;4 FAG/B,yB;MAAA,2D;MAAA,+D;MAAA,sC;QAYc,Q;QAFV,UAAU,c;QACV,WAAW,gB;QACD,2B;QAAV,O AAU,cAAV,C;UAAU,mB;UACN,UAAU,SAAS,CAAT,C;UACV,IAAI,GAAL,WAAI,GAAL,CAAR,C;YACI,IAA K,WAAI,CAAJ,C;;QAEb,OAAO,I;O;KAjBX,C;IAoBA,uC;MAQI,UAAe,eAAL,SAAK,C;MACX,YAAJ,GAAL,E AAU,KAAV,C;MACJ,OAAO,G;K;IAGX,sC;MAMI,UAAe,eAAL,SAAK,C;MACX,YAAJ,GAAL,EAAU,KAAV, C;MACJ,OAAO,G;K;IAGX,mC;MAMiB,IAAN,I;MACH,kBADs,SACT,c;QAAoB,4BAAc,SAAd,C;;QACZ,iCA Aa,sBAAb,C;MAFZ,W;K;IAMJ,mC;MAUI,UAAe,eAAL,SAAK,C;MACX,OAAJ,GAAL,EAAO,KAAP,C;MACJ, OAAO,G;K;8EAGX,yB;MAAA,gD;MAAA,uC;QAOoB,Q;QADhB,IAAI,wCAAsB,mBAA1B,C;UAAqC,OAAO,I ;QAC5B,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UAAM,IAAI,CAAC,UAAU,OAAV,CAAL,C;YAAyB,OAA O,K;;QACtD,OAAO,I;O;KARX,C;IAWA,2B;MAMI,IAAI,oCAAJ,C;QAAwB,OAAO,CAAC,mB;MACHc,OAAO ,oBAAW,U;K;+EAGtB,yB;MAAA,gD;MAAA,uC;QAOoB,Q;QADhB,IAAI,wCAAsB,mBAA1B,C;UAAqC,OAA O,K;QAC5B,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UAAM,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,I; ;QACrD,OAAO,K;O;KARX,C;IAWA,6B;MAMoB,Q;MAFhB,IAAI,oCAAJ,C;QAAwB,OAAO,c;MAC/B,YAAy, C;MACI,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,oBAAmB,qBAAnB,EAAMB,KAAAnB,E;;MACtB,O AAO,K;K;mFAGX,qB;MAKI,OAAO,c;K;mFAGX,yB;MAAA,gD;MAAA,wE;MAAA,uC;QAMoB,Q;QAFhB,IA AI,wCAAsB,mBAA1B,C;UAAqC,OAAO,C;QAC5C,YAAy,C;QACI,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB; UAAM,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,oBAAmB,qBAAnB,EAAMB,KAAAnB,E;;QAC9C,OAAO,K;O;KA PX,C;gFAUA,yC;MAUoB,Q;MADhB,kBAAkB,O;MACF,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,cA Ac,UAAU,WAAV,EAAB,UAAV,C;;MACpC,OAAO,W;K;8FAGX,yB;MAAA,wE;MAAA,gD;QAYoB,UAAiD ,M;QAFjE,YAAy,C;QACZ,kBAAkB,O;QACF,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UAAM,cAAc,UAAU, oBAAMb,cAAAnB,EAAMb,sBAAnB,UAAV,EAAB,cAAc,UAAV,C;;QACpC,OAAO,W;O;KAbX,C ;0FAGBA,yC;MASI,kBAAkB,O;MACIB,IAAI,CAAC,mBAAL,C;QACI,eAAe,+BAAa,cAAb,C;QACf,OAAO,QA AS,cAAhB,C;UACI,cAAc,UAAU,QAAS,WAAAnB,EAAB,WAA/B,C;;OAGtB,OAAO,W;K;wGAGX,yC;MAUI,k BAAkB,O;MACIB,IAAI,CAAC,mBAAL,C;QACI,eAAe,+BAAa,cAAb,C;QACf,OAAO,QAAS,cAAhB,C;UACI,Y AAY,QAAS,gB;UACrB,cAAc,UAAU,KAAV,EAAB,QAAS,WAA1B,EAAsC,WAAiC,C;;OAGtB,OAAO,W;K;s FAGX,6B;MAKoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,OAAO,OAAP,C;;K;oGAG1B,y B;MAAA,wE;MAAA,oC;QAOiB,UAAgC,M;QAD7C,YAAy,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UAA M,OAAO,oBAAMb,cAAAnB,EAAMb,sBAAnB,UAAP,EAAB,cAAc,IAAP,C,C;;O;KAPvB,C;IAUA,0B;MAII,OAAO,sB ;K;IAGX,2B;MAII,OAAO,uB;K;IAGX,2B;MAGI,OAAO,uB;K;kFAGX,+B;MAGW,sB;;QAUP,eAAe,oB;QACf,I AAI,CAAC,QAAS,UAAc,C;UAAyB,qBAAO,I;UAAp,uB;SACzB,cAAc,QAAS,O;QACvB,IAAI,CAAC,QAAS,U AAd,C;UAAyB,qBAAO,O;UAAp,uB;SACzB,eAdmB,QAcJ,CAAS,OAAT,C;;UAEX,QAAQ,QAAS,O;UACjB,Q AjBe,QAiBP,CAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAED, QAAT,QAAS,W;QACIB,qBAAO,O;;MAvBP,yB;K;8FAGJ,+B;MAOI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAA d,C;QAAyB,OAAO,I;MACHc,cAAc,QAAS,O;MACvB,IAAI,CAAC,QAAS,UAAc,C;QAAyB,OAAO,O;MACHc, eAAe,SAAS,OAAT,C;;QAEX,QAAQ,QAAS,O;QACjB,QAAQ,SAAS,CAAT,C;QACR,IAAI,2BAAW,CAAX,KA AJ,C;UACI,UAAU,C;UACV,WAAW,C;;MAED,QAAT,QAAS,W;MACIB,OAAO,O;K;mFAGX,yB;MAAA,sE;M F/yDA,iB;ME+yDA,sC;QAaI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAc,C;UAAyB,MAAM,6B;QAC/B,eAAe,S AAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,WFzzDG,M AAO,KEyzDO,QFzzDP,EEyzDiB,CFzzDjB,C;;QE2zDd,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MFj1DA ,iB;MEi1DA,sC;QAaI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAc,C;UAAyB,MAAM,6B;QAC/B,eAAe,SAAS,Q AAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,WF31DG,MAAO, KE21DO,QF31DP,EE21DiB,CF31DjB,C;;QE61Dd,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MAAA,sC;Q AWI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAc,C;UAAyB,MAAM,6B;QAC/B,eAAe,SAAS,QAAS,OAAIB,C;Q

ACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YAC I,WAAW,C;;QAGnB,OAAO,Q;O;KApBX,C;+FAuBA,yB;MFp3DA,iB;MEo3DA,sC;QAWI,eAAe,oB;QACf,IAAI ,CAAC,QAAS,UAAAd,C;UAAyB,OAAO,I;QAChC,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C ;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,WF53DG,MAAO,KE43DO,QF53DP,EE43DiB,CF53DjB,C;;QE83 Dd,OAAO,Q;O;KAIBX,C;+FAqBA,yB;MFp5DA,iB;MEo5DA,sC;QAWI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UA Ad,C;UAAyB,OAAO,I;QAChC,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SA AS,QAAS,OAAIB,C;UACR,WF55DG,MAAO,KE45DO,QF55DP,EE45DiB,CF55DjB,C;;QE85Dd,OAAO,Q;O;K AIBX,C;+FAqBA,+B;MASI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHC,eAAe,SAA S,QAAS,OAAIB,C;MACf,OAAO,QAAS,UAAhB,C;QACI,QAAQ,SAAS,QAAS,OAAIB,C;QACR,IAAI,2BAAW, CAAX,KAAJ,C;UACI,WAAW,C;;MAGnB,OAAO,Q;K;0FAGX,yB;MAAA,sE;MAAA,kD;QAWI,eAAe,oB;QAC f,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;QAC/B,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS, UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GA AkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KApBX,C;sGAuBA,2C;MASI,eAAe,oB;MACf,IAAI,CAA C,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHC,eAAe,SAAS,QAAS,OAAIB,C;MACf,OAAO,QAAS,UAAhB,C;QA CI,QAAQ,SAAS,QAAS,OAAIB,C;QACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C ;UACI,WAAW,C;;MAGnB,OAAO,Q;K;IAGX,gC;MAOI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAyB, OAAO,I;MACHC,UAAU,QAAS,O;MACnB,OAAO,QAAS,UAAhB,C;QACI,QAAQ,QAAS,O;QACjB,MFn+DG, MAAO,KEm+DE,GFn+DF,EE+DO,CFn+DP,C;;MEq+Dd,OAAO,G;K;IAGX,iC;MAOI,eAAe,oB;MACf,IAAI,C AAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHC,UAAU,QAAS,O;MACnB,OAAO,QAAS,UAAhB,C;QACI,QAA Q,QAAS,O;QACjB,MF//DG,MAAO,KE+/DE,GF//DF,EE+/DO,CF//DP,C;;MEigEd,OAAO,G;K;IAGX,iC;MAKI,e AAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHC,UAAU,QAAS,O;MACnB,OAAO,QAAS, UAAhB,C;QACI,QAAQ,QAAS,O;QACjB,IAAI,sBAAM,CAAN,KAAJ,C;UAAa,MAAM,C;;MAEvB,OAAO,G;K ;IAGX,0C;MAGI,OAAO,2BAAc,UAAAd,C;K;IAGX,gD;MAKI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QA AyB,OAAO,I;MACHC,UAAU,QAAS,O;MACnB,OAAO,QAAS,UAAhB,C;QACI,QAAQ,QAAS,O;QACjB,IAAI, UAAW,SAAQ,GAAR,EAaA,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IAGX,0 B;MAIL,OAAO,sB;K;IAGX,2B;MAIL,OAAO,uB;K;IAGX,2B;MAGI,OAAO,uB;K;kFAGX,+B;MAGW,sB;;QAUP ,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,I;UAAP,uB;SACzB,cAAc,QAAS,O;QACvB,IAA I,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,O;UAAP,uB;SACzB,eAdmB,QAcJ,CAAS,OAAT,C;;UAEX,QAAQ,Q AAS,O;UACjB,QAjBe,QAiBP,CAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,W AAW,C;;QAED,QAAT,QAAS,W;QACIB,qBAAO,O;;MAvBP,yB;K;8FAGJ,+B;MAOI,eAAe,oB;MACf,IAAI,CA AC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHC,cAAc,QAAS,O;MACvB,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,O AAO,O;MACHC,eAAe,SAAS,OAAT,C;;QAEX,QAAQ,QAAS,O;QACjB,QAAQ,SAAS,CAAT,C;QACR,IAAI,2B AAW,CAAX,KAAJ,C;UACI,UAAU,C;UACV,WAAW,C;;MAED,QAAT,QAAS,W;MACIB,OAAO,O;K;mFAGX ,yB;MAAA,sE;MF14DA,iB;MEk4DA,sC;QAaI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B ;QAC/B,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;U ACR,WF54DG,MAAO,KE44DO,QF54DP,EE44DiB,CF54DjB,C;;QE84Dd,OAAO,Q;O;KApBX,C;mFAuBA,yB; MAAA,sE;MFp6DA,iB;MEo6DA,sC;QAaI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;Q AC/B,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UAC R,WF96DG,MAAO,KE86DO,QF96DP,EE86DiB,CF96DjB,C;;QEg7Dd,OAAO,Q;O;KApBX,C;mFAuBA,yB;MA AA,sE;MAAA,sC;QAWI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;QAC/B,eAAe,SAAS, QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,IAAI,2BAAW,C AAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KApBX,C;+FAuBA,yB;MFv8DA,iB;MEu8DA,sC;QAWI,e AAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,OAAO,I;QAChC,eAAe,SAAS,QAAS,OAAIB,C;QACf,OA AO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,WF/8DG,MAAO,KE+8DO,QF/8DP,EE+8Di B,CF/8DjB,C;;QEi9Dd,OAAO,Q;O;KAIBX,C;+FAqBA,yB;MFv+DA,iB;MEu+DA,sC;QAWI,eAAe,oB;QACf,IA AI,CAAC,QAAS,UAAAd,C;UAAyB,OAAO,I;QAChC,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAh B,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,WF/+DG,MAAO,KE++DO,QF/+DP,EE++DiB,CF/+DjB,C;;QE i/Dd,OAAO,Q;O;KAIBX,C;+FAqBA,+B;MASI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;

MACHc,eAAe,SAAS,QAAS,OAAIB,C;MACf,OAAO,QAAS,UAAhB,C;QACI,QAAQ,SAAS,QAAS,OAAIB,C;QACR,IAAI,2BAAW,CAAX,KAJ,C;UACI,WAAW,C;;MAGnB,OAAO,Q;K;0FAGX,yB;MAAA,sE;MAAA,kD;QAWI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;QAC/B,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KApBX,C;sGAuBA,2C;MASI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAYB,OAAO,I;MACHc,eAAe,SAAS,QAAS,OAAIB,C;MACf,OAAO,QAAS,UAAhB,C;QACI,QAAQ,SAAS,QAAS,OAAIB,C;QACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;UACI,WAAW,C;;MAGnB,OAAO,Q;K;IAGX,gC;MAOI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAYB,OAAO,I;MACHc,UAAU,QAAS,O;MACnB,OAAO,QAAS,UAAhB,C;QACI,QAAQ,QAAS,O;QACjB,MFtjEG,MAAO,KEsjEE,GFtjEF,EEsjEO,CFtjEP,C;;MEwjEd,OAAO,G;K;IAGX,iC;MAOI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAYB,OAAO,I;MACHc,UAAU,QAAS,O;MACnB,OAAO,QAAS,UAAhB,C;QACI,QAAQ,QAAS,O;QACjB,MFIIEG,MAAO,KEkIEE,GFIIIE,EEkIEO,CFIIIEP,C;;MEolEd,OAAO,G;K;IAGX,iC;MAKI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAYB,OAAO,I;MACHc,UAAU,QAAS,O;MACnB,OAAO,QAAS,UAAhB,C;QACI,QAAQ,QAAS,O;QACjB,IAAI,sBAAM,CAAN,KAJ,C;UAAa,MAAM,C;;MAEvB,OAAO,G;K;IAGX,0C;MAGI,OAAO,2BAAc,UAAAd,C;K;IAGX,gD;MAKI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAYB,OAAO,I;MACHc,UAAU,QAAS,O;MACnB,OAAO,QAAS,UAAhB,C;QACI,QAAQ,QAAS,O;QACjB,IAAI,CAAC,QAAS,UAAAd,C;K;IAGX,4B;MAMI,IAAI,oCAAJ,C;QAawB,OAAO,mB;MAC/B,OAAO,CAAC,oBAAW,U;K;iFAGvB,yB;MAAA,gD;MAAA,uC;QAOoB,Q;QADhB,IAAI,wCAAsB,mBAA1B,C;UAAqC,OAAO,I;QAC5B,2B;QAahB,OAAgB,cAAhB,C;UAAgB,yB;UAAM,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,K;;QACrD,OAAO,I;O;KARX,C;oFAWA,6B;MAKmC,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAGB,yB;QAAM,OAAO,OAAP,C;;MAArC,gB;K;kGAGJ,yB;MAAA,6B;MAAA,sC;MArnBA,wE;MAqnBA,2BAQiB,yB;QA7nBjB,wE;eA6nBiB,0B;UAAA,4B;YAAE,aAAe,c;YAtmBjB,gB;YADb,YAAY,C;YACC,2B;YAAb,OAAa,cAAb,C;CAAA,sB;cAAM,OAAO,oBAAmB,cAAnB,EAAMb,sBAAnB,UAAP,EAAC,IAApC,C;;YAsnBmB,W;W;S;OAAzB,C;MARjB,oC;QA9mBiB,gB;QADb,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UAAM,OAAO,oBAAmB,cAAnB,EAAMb,sBAAnB,UAAP,EAAC,IAApC,C;;QAsnBnB,gB;O;KARJ,C;oFAWA,yB;MAAA,4F;MAAA,uC;QAaI,eAAe,SAAK,W;QACpB,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,mCAA8B,oCAA9B,C;QAC/B,kBAAqB,QAAS,O;QAC9B,OAAO,QAAS,UAAhB,C;UACI,cAAc,UAAU,WAAV,EAauB,QAAS,OAAhC,C;;QAEIB,OAAO,W;O;KAnBX,C;kGAsBA,yB;MAAA,4F;MAAA,wE;MAAA,uC;QAKBmD,Q;QAL/C,eAAe,SAAK,W;QACpB,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,mCAA8B,oCAA9B,C;QAC/B,YAAY,C;QACZ,kBAAqB,QAAS,O;QAC9B,OAAO,QAAS,UAAhB,C;UACI,cAAc,UAAU,oBAAmB,YAAnB,EAAMb,oBAAnB,QAAY,EAAC,WAAvC,EAAD,QAAS,OAA7D,C;;QAEIB,OAAO,W;O;KApBX,C;8GAuBA,yB;MAAA,wE;MAAA,uC;QAKBmD,Q;QAL/C,eAAe,SAAK,W;QACpB,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,OAAO,I;QACHc,YAAY,C;QACZ,kBAAqB,QAAS,O;QAC9B,OAAO,QAAS,UAAhB,C;UACI,cAAc,UAAU,oBAAmB,YAAnB,EAAMb,oBAAnB,QAAY,EAAC,WAAvC,EAAD,QAAS,OAA7D,C;;QAEIB,OAAO,W;O;KApBX,C;gGAuBA,gC;MAcI,eAAe,SAAK,W;MACpB,IAAI,CAAC,QAAS,UAAAd,C;QAAYB,OAAO,I;MACHc,kBAAqB,QAAS,O;MAC9B,OAAO,QAAS,UAAhB,C;QACI,cAAc,UAAU,WAAV,EAauB,QAAS,OAAhC,C;;MAEIB,OAAO,W;K;8FAGX,yB;MAAA,4F;MAAA,uC;QAaI,eAAe,+BAAa,cAAb,C;QACf,IAAI,CAAC,QAAS,cAAAd,C;UACI,MAAM,mCAA8B,8BAA9B,C;QACV,kBAAqB,QAAS,W;QAC9B,OAAO,QAAS,cAAhB,C;UACI,cAAc,UAAU,QAAS,WAAAnB,EAAB,WAA/B,C;;QAEIB,OAAO,W;O;KApBX,C;4GAuBA,yB;MAAA,4F;MAAA,uC;QAaI,eAAe,+BAAa,cAAb,C;QACf,IAAI,CAAC,QAAS,cAAAd,C;UACI,MAAM,mCAA8B,8BAA9B,C;QACV,kBAAqB,QAAS,W;QAC9B,OAAO,QAAS,cAAhB,C;UACI,YAAY,QAAS,gB;UACrB,cAAc,UAAU,KAAY,EAAB,WAA/B,C;;QAEIB,OAAO,W;O;KArBX,C;wHAWBA,gC;MAaI,eAAe,+BAAa,cAAb,C;MACf,IAAI,CAAC,QAAS,cAAAd,C;QACI,OAAO,I;MACX,kBAAqB,QAAS,W;MAC9B,OAAO,QAAS,cAAhB,C;QACI,YAAY,QAAS,gB;QACrB,cAAc,UAAU,KAAY,EAAB,WAA/B,C;;MAEIB,OAAO,W;K;0GAGX,gC;MAcI,eAAe,+BAAa,cAAb,C;MACf,IAAI,CAAC,QAAS,cAAAd,C;QACI,OAAO,I;MACX,kBAAqB,QAAS,W;MAC9B,OAAO,QAAS,cAAhB,C;QACI,cAAc,UAAU,QAAS,WAAAnB,EAAB,WAA/B,C;;MAEIB,OAAO,W;K;8FAGX,yB;MAAA,kF;MAAA,gD;MAAA,gE;MAAA,gD;QAiBoB,Q;QAJhB,oBAAoB,mCAAwB,CAAxB,C;QACpB,IAAI,kBAAiB,CAArB,C;UAAwB,OAAO

,OAAO,OAAP,C;QACc,kBAAhC,eAAa,gBAAgB,CAAhB,IAAb,C;QAAwC,8B;QAArD,aHjjFO,W;QGkjFP,kBA AkB,O;QACF,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;UACd ,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KArBX,C;4GAwBA,yB;MAAA,kF;MAAA,gD;MAAA,gE;MAAA, gD;QAmBoB,UACY,M;QAN5B,oBAAoB,mCAAwB,CAAxB,C;QACpB,IAAI,kBAAiB,CAArB,C;UAAwB,OAA O,OAAO,OAAP,C;QACc,kBAAhC,eAAa,gBAAgB,CAAhB,IAAb,C;QAAwC,8B;QAArD,aH1kFO,W;QG2kFP,Y AAY,C;QACZ,kBAAkB,O;QACF,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,cAAc,WAAU,cAAV,EAAU ,sBAAV,WAAmB,WAAmB,EAAgC,OAAhC,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAvBX,C;k GA0BA,yB;MAAA,qD;MAAA,kF;MAAA,gE;MAAA,uC;QAcI,eAAe,SAAK,W;QACpB,IAAI,CAAC,QAAS,UA Ad,C;UAAyB,OAAO,W;QACc,sBAAqB,QAAS,OAA9B,C;QACuD,kBAA1C,eAAa,mCAAwB,EAAxB,CAAb, C;QAAkD,sBAAI,aAAJ,C;QAA/D,aHrmFO,W;QGsmFP,OAAO,QAAS,UAAhB,C;UACI,gBAAc,UAAU,aAAV, EAAuB,QAAS,OAAhC,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAtBX,C;gHAYBA,yB;MAAA,qD ;MAAA,kF;MAAA,gE;MAAA,uC;QAOBgC,Q;QAN5B,eAAe,SAAK,W;QACpB,IAAI,CAAC,QAAS,UAAc,C;U AAYB,OAAO,W;QACc,sBAAqB,QAAS,OAA9B,C;QACuD,kBAA1C,eAAa,mCAAwB,EAAxB,CAAb,C;QAA kD,sBAAI,aAAJ,C;QAA/D,aH9nFO,W;QG+nFP,YAAY,C;QACZ,OAAO,QAAS,UAAhB,C;UACI,gBAAc,WAA U,YAAV,EAAU,oBAAV,SAAmB,aAAmB,EAAgC,QAAS,OAAzC,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAA O,M;O;KAvBX,C;gFA0BA,yB;MarGA,kF;MAAA,gD;MAAA,gE;MAqGA,gD;QAcW,sB;;UAIGS,Q;UAJhB,oB AAoB,mCAAwB,CAAxB,C;UACpB,IAAI,kBAAiB,CAArB,C;YAAwB,qBAAO,OAqGZ,OAqGY,C;YAAP,uB;W ACqB,kBAAhC,eAAa,gBAAgB,CAAhB,IAAb,C;UAAwC,sBAoGIC,OAqGkC,C;UAArD,aHjjFO,W;UGkjFP,kB AmGmB,O;UAIGH,2B;UAAhB,OAAGB,cAAhB,C;YAAgB,yB;YACZ,cAiGwB,SAjGV,CAAU,WAAV,EAAuB, OAAvB,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;QA8FP,yB;O;KAdJ,C;8FAiBA,yB;MA9FA,kF;M AAA,gD;MAAA,gE;MA8FA,gD;QAeW,6B;;UA1FS,gB;UALhB,oBAAoB,mCAAwB,CAAxB,C;UACpB,IAAI,k BAAiB,CAArB,C;YAAwB,4BAAO,OA8FL,OA9FK,C;YAAP,8B;WACqB,kBAAhC,eAAa,gBAAgB,CAAhB,IA Ab,C;UAAwC,sBA6F3B,OA7F2B,C;UAArD,aH1kFO,W;UG2kFP,YAAY,C;UACZ,kBA2F0B,O;UA1FV,2B;UA AhB,OAAGB,cAAhB,C;YAAgB,yB;YACZ,cAyF+B,SAzFjB,EAAU,cAAV,EAAU,sBAAV,WAAmB,WAAmB,EA AgC,OAAhC,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,M;;QAsFP,gC;O;KafJ,C;kFAkBA,+B;MAOoB, Q;MADhB,UAAe,C;MACC,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,YAAO,SAAS,OAAT,CAAP,I;;M AEJ,OAAO,G;K;8FAGX,+B;MAOoB,Q;MADhB,UAAkB,G;MACF,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB; QACZ,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;mFAGX,+B;MAUoB,Q;MADhB,UAAoB,C;MACJ,2B;MAA hB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;mFAGX,+B;MAUoB,Q;M ADhB,UAAe,C;MACC,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,YAAO,SAAS,OAAT,CAAP,I;;MAEJ, OAAO,G;K;mFAGX,yB;MAAA,SASoB,gB;MATpB,sC;QAUoB,Q;QADhB,Y;QACgB,2B;QAAhB,OAAGB,cAA hB,C;UAAgB,yB;UACZ,cAAO,SAAS,OAAT,CAAP,C;;QAEJ,OAAO,G;O;KAbX,C;mFAgBA,yB;MjB/7EA,6B; MiB+7EA,sC;QAWoB,Q;QADhB,UjB/7EmC,ciB+7EnB,CjB/7EmB,C;QiBg8EnB,2B;QAAhB,OAAGB,cAAhB,C; UAAgB,yB;UACZ,MjBnwFiD,ciBmwFjD,GjBnwF2D,KAAK,GiBmwFzD,SAAS,OAAT,CjBnwFoE,KAAK,IAAf ,C;;QiBqwFrD,OAAO,G;O;KAdX,C;mFAiBA,yB;MD78EA,+B;MC68EA,sC;QAWoB,Q;QADhB,UD58EqC,eAA W,oBC48E/B,CD58E+B,CAAX,C;QC68ErB,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,MDjxFmD,eCixF nD,GDjxF8D,KAAK,KCixF5D,SAAS,OAAT,CDjxFuE,KAAK,CAAhB,C;;QCmxFvD,OAAO,G;O;KAdX,C;IAiB A,qC;MAIoB,UAMT,M;MANS,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,IAAI,eAAJ,C;UACI,MAAM, gCAAYB,2BAAwB,SAAXB,MAAZB,C;;MAId,OAAO,mE;K;IAGX,qC;MAIoB,UAMT,M;MANS,2B;MAAhB,O AAGB,cAAhB,C;QAAGB,yB;QACZ,IAAI,eAAJ,C;UACI,MAAM,gCAAYB,2BAAwB,SAAXB,MAAZB,C;;MAId, OAAO,+D;K;IAGX,kC;MAWI,OAAO,oBAAS,IAAT,EAAe,IAAf,EAAc,IAAtC,C;K;IAGX,+C;MAGBI,OAAO,s BAAS,IAAT,EAAe,IAAf,EAAc,IAAtC,EAAwD,SAAXD,C;K;IAGX,mC;MAII,aAAa,iBAAa,mCAAwB,EAAxB ,CAAb,C;MACb,kBAAc,KAAd,C;MANlEgB,Q;MAAA,OAoLET,SAPLES,W;MAAhB,OAAGB,cAAhB,C;QAAGB, 2B;QAAU,oB;QAOIEK,IAAI,CAAC,SAAD,IAAY,OAplEX,SAoIEW,UAAhB,C;UAAiC,YAAU,I;UAA3C,mBAA iD,K;;UAAjD,mBAA8D,I;;QAplEvE,qB;UAoIED,MAplEqC,WAAI,SAAJ,C;;MAoEID,OAAqB,M;K;IAGzB,sC; MAQI,IAAI,QpB0yJG,YAAQ,CoB1yJf,C;QAAwB,OAAy,SAAL,SAAK,C;MACpC,YAAqB,8BAAT,QAAS,C;M AtoEd,kBAAy,gB;MA4BH,Q;MAAA,OA2mET,SA3mES,W;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,IA AI,CA2mEF,qBA3mEa,OA2mEb,CA3mEF,C;UAAyB,WAAy,WAAI,OAAJ,C;;MA2mE3D,OA1mEO,W;K;IA6m

EX,sC;MAQI,YAAqB,gCAAT,QAAS,EAAgC,SAAhC,C;MACrB,IAAI,KAAM,UAAV,C;QACI,OAAy,SAAL,S  
AAK,C;MAppET,kBAAY,gB;MA4BH,Q;MAAA,OAYnET,SAznES,W;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;Q  
AAM,IAAI,CAynEF,qBAznEa,OAYnEb,CAznEF,C;UAAyB,WAAy,WAAI,OAAJ,C;;MAynE3D,OAXnEO,W;K;I  
A2nEX,sC;MAQI,YAAqB,8BAAT,QAAS,C;MACrB,IAAI,KAAM,UAAV,C;QACI,OAAy,SAAL,SAAK,C;MAI  
qET,kBAAY,gB;MA4BH,Q;MAAA,OAuoET,SAvoES,W;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,IAAI,  
CAuoEF,qBAvoEa,OAuoEb,CAvoEF,C;UAAyB,WAAy,WAAI,OAAJ,C;;MAuoE3D,OAtEO,W;K;8FAyoEX,yB  
;MAAA,8C;MAAA,qC;QAKI,OAAO,iBAAM,OAAN,C;O;KALX,C;0FAQA,yB;MAAA,+D;MAAA,6B;MAAA,u  
C;QAUoB,Q;QAFhB,YAAy,gB;QACZ,aAAa,gB;QACG,2B;QAAhB,OAAGB,cAAhB,C;UAGB,yB;UACZ,IAAI  
,UAAU,OAAV,CAAJ,C;YACI,KAAM,WAAI,OAAJ,C;;YAEN,MAAO,WAAI,OAAJ,C;;;QAGf,OAAO,cAAK,K  
AAL,EAAy,MAAZ,C;O;KAjBX,C;IAoBa,kC;MAII,IAAI,oCAAJ,C;QAawB,OAAy,OAAL,SAAK,EAAK,OAA  
L,C;MACpC,aAAa,gB;MACN,OAAP,MAAO,EAAO,SAAP,C;MACP,MAAO,WAAI,OAAJ,C;MACP,OAAO,M;  
K;IAGX,oC;MAII,aAAa,iBAaA,iBAAO,CAAP,IAAb,C;MACb,MAAO,gBAAO,SAAP,C;MACP,MAAO,WAAI,  
OAAJ,C;MACP,OAAO,M;K;IAGX,qC;MAII,IAAI,oCAAJ,C;QAawB,OAAy,OAAL,SAAK,EAAK,QAAL,C;M  
ACpC,aAAa,gB;MACN,OAAP,MAAO,EAAO,SAAP,C;MACA,SAAP,MAAO,EAAO,QAAP,C;MACP,OAAO,M  
;K;IAGX,qC;MAII,aAAa,iBAaA,SAAK,KAAL,GAAY,QAAS,OAARb,IAAb,C;MACb,MAAO,gBAAO,SAAP,C;  
MACA,SAAP,MAAO,EAAO,QAAP,C;MACP,OAAO,M;K;IAGX,qC;MAII,IAAI,oCAAJ,C;QAawB,OAAy,OA  
AL,SAAK,EAAK,QAAL,C;MACpC,aAAa,gB;MACN,OAAP,MAAO,EAAO,SAAP,C;MACA,OAAP,MAAO,EA  
AO,QAAP,C;MACP,OAAO,M;K;IAGX,qC;MAII,IAAI,mCAAJ,C;QACI,aAAa,iBAaA,SAAK,KAAL,GAAY,QA  
AS,KAARb,IAAb,C;QACb,MAAO,gBAAO,SAAP,C;QACP,MAAO,gBAAO,QAAP,C;QACP,OAAO,M;;QAEP,e  
AAa,iBAaA,SAAb,C;QACN,OAAP,QAAO,EAAO,QAAP,C;QACP,OAAO,Q;;K;IAIf,qC;MAII,aAAa,gB;MACN,  
OAAP,MAAO,EAAO,SAAP,C;MACA,SAAP,MAAO,EAAO,QAAP,C;MACP,OAAO,M;K;IAGX,qC;MAII,aAA  
a,iBAaA,SAAK,KAAL,GAAY,EAAZ,IAAb,C;MACb,MAAO,gBAAO,SAAP,C;MACA,SAAP,MAAO,EAAO,Q  
AAP,C;MACP,OAAO,M;K;4FAGX,yB;MAAA,4C;MAAA,qC;QAKI,OAAO,gBAAK,OAAL,C;O;KALX,C;8FA  
QA,yB;MAAA,4C;MAAA,qC;QAKI,OAAO,gBAAK,OAAL,C;O;KALX,C;IAQA,yD;MAGB+C,oB;QAAA,OAA  
Y,C;MAAG,8B;QAAA,iBAA0B,K;MAOzE,Q;MANX,oBAAoB,IAApB,EAA0B,IAA1B,C;MACA,IAAI,0CAAw  
B,8BAA5B,C;QACI,eAAe,SAAK,K;QACpB,qBAAqB,YAAW,IAAX,SAASB,WAAW,IAAX,KAAMB,CAAvB,G  
AA0B,CAA1B,GAAiC,CAAnD,K;QACrB,aAAa,iBAAmB,cAAAnB,C;QACb,gBAAY,CAAZ,C;QACA,Y;UAAO,c  
;UAAP,MAAGB,CAAT,mBAAiB,QAAxB,E;YAAA,K;UACI,iBAASB,eAAL,IAAK,EAAa,WAAW,OAAx,IAAb,  
C;UACtB,IAAI,aAAa,IAAb,IAAqB,CAAC,cAA1B,C;YAA0C,K;Ud59FID,WAAW,iBc69Fa,Ud79Fb,C;UWCX,m  
BAAc,CAAd,YG49FwB,UH59FxB,Y;YXA6B,ec49FS,sBH39F3B,OG29FgC,GAAK,OAAL,IAAL,Cd59FT,C;;Uc  
49FrB,MAAO,Wd39FR,Ic29FQ,C;UACP,oBAAS,IAAT,I;;QAEJ,OAAO,M;OAEX,eAAa,gB;MACiE,kBAA9E,iB  
AAiB,oBAAjB,EAA6B,IAA7B,EAAmC,IAAnC,EAAyC,cAAzC,EAAuE,IAAvE,C;ME51GA,OAAGB,qBAAhB,  
C;QAAGB,gC;QF6lGL,mBE7lGqB,OF6lGrB,C;;MAEX,OAAO,Q;K;IAGX,sE;MAkBkD,oB;QAAA,OAAy,C;MA  
AG,8B;QAAA,iBAA0B,K;MACvF,oBAAoB,IAApB,EAA0B,IAA1B,C;MACA,IAAI,0CAAwB,8BAA5B,C;QACI  
,eAAe,SAAK,K;QACpB,qBAAqB,YAAW,IAAX,SAASB,WAAW,IAAX,KAAMB,CAAvB,GAA0B,CAA1B,GAA  
iC,CAAnD,K;QACrB,aAAa,iBAaA,cAAb,C;QACb,eAAa,kBAAc,SAAd,C;QACb,YAAy,C;QACZ,OAAGB,CAA  
T,qBAAiB,QAAxB,C;UACI,iBAASB,eAAL,IAAK,EAAa,WAAW,KAAX,IAAb,C;UACtB,IAAI,CAAC,cAAD,IA  
AmB,aAAa,IAApC,C;YAA0C,K;UAC1C,QAAO,cAAK,KAAL,EAAy,QAQQ,UAAAR,IAAZ,C;UACP,MAAO,W  
AAI,UAAU,QAAV,CAAJ,C;UACP,gBAAS,IAAT,I;;QAEJ,OAAO,M;OAEX,eAAa,gB;MACgE,kBAA7E,iBAAi  
B,oBAAjB,EAA6B,IAA7B,EAAmC,IAAnC,EAAyC,cAAzC,EAAuE,IAAvE,C;MEtoGA,OAAGB,qBAAhB,C;QA  
AgB,gC;QFuoGL,mBAAI,UEvoGiB,OFuoGjB,CAAJ,C;;MAEX,OAAO,Q;K;IAGX,kC;MAqBoB,gB;MAHhB,gB  
AXW,KAWW,O;MACtB,WAAW,iBF17FJ,MAAO,KE07FgB,mCAAwB,EAAxB,CF17FhB,EE07F6C,SF17F7C,C  
E07FH,C;MACX,QAAQ,C;MACQ,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,IAAI,KAAC,SAAT,C;UA  
AoB,K;QACpB,IAAK,WAhBqB,GAgBP,OAHO,EAAnB,KAgBqB,CAAM,UAAAN,EAAM,kBAAN,SAhBF,CAG  
BrB,C;;MAhBT,OAkBO,I;K;+EafX,yB;MAAA,kF;MAAA,gE;MFv7FA,iB;MEu7FA,8C;QAWoB,UAEsB,M;QA  
LtC,gBAAGB,KAAM,O;QACtB,WAAW,eF17FJ,MAAO,KE07FgB,mCAAwB,EAAxB,CF17FhB,EE07F6C,SF17  
F7C,CE07FH,C;QACX,QAAQ,C;QACQ,2B;QAAhB,OAAGB,cAAhB,C;UAGB,yB;UACZ,IAAI,KAAC,SAAT,  
C;YAAoB,K;UACpB,IAAK,WAAI,UAAU,OAAV,EAAMB,MAAM,UAAAN,EAAM,kBAAN,SAANB,CAAJ,C;;Q



,CAAT,EAAkC,UAAIC,C;K;IAGX,oC;MAOI,OAAW,UAAW,SAAQ,CAAR,EAAW,CAAX,CAAX,IAA4B,CAA  
hC,GAAmC,CAAnC,GAA0C,C;K;IAmDrD,wC;MAQc,Q;MADV,UAAU,C;MACV,wBAAU,KAAV,gB;QAAU,Q  
AAA,KAAV,M;QAAiB,IAAI,UAAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;  
MAC3D,OAAO,G;K;oGCnXX,yB;MAAA,iE;MAAA,uC;QASW,Q;QAAA,+B;;UAYS,U;UAAA,SjB4UoE,iBAA  
Q,W;UiB5U5F,OAAgB,gBAAhB,C;YAAgB,2B;YACZ,aAbwB,SAaX,CAAU,OAAV,C;YACb,IAAI,CAAJ,C;cAC  
I,8BAAO,M;cAAP,gC;;UAGR,8BAAO,I;;QAlBA,kC;QAAA,iB;UAAmC,MAAM,gCAAuB,4DAAvB,C;SAAhD,  
OAAO,I;O;KATX,C;gHAYA,gC;MASoB,Q;MAAA,OAAA,SjB4UoE,QAAQ,W;MiB5U5F,OAAgB,cAAhB,C;Q  
AAgB,yB;QACZ,aAAa,UAAU,OAAV,C;QACb,IAAI,CAAJ,C;UACI,OAAO,M;;MAGf,OAAO,I;K;IAGX,6B;MA  
II,IAAI,mBAAQ,CAAZ,C;QACI,OAAO,W;MACX,eAAe,iBAAQ,W;MACvB,IAAI,CAAC,QAAS,UAAAd,C;QAC  
I,OAAO,W;MACX,YAAY,QAAS,O;MACrB,IAAI,CAAC,QAAS,UAAAd,C;QACI,OAAO,OjB8PiD,SiB9P1C,KjB  
8P+C,IAAL,EiB9P1C,KjB8PoD,MAAV,CiB9PjD,C;OACX,aAAa,iBAAsB,cAAtB,C;MACb,MAAO,WjB4PqD,Si  
B5PjD,KjB4PsD,IAAL,EiB5PjD,KjB4P2D,MAAV,CiB5PrD,C;;QAEwB,kBAAhB,QAAS,O;QAApB,MAAO,WjB  
0PiD,SAAK,eAAL,EAAU,iBAAV,CiB1PjD,C;;MACO,QAAT,QAAS,W;MACiB,OAAO,M;K;uFAGX,yB;MAAA  
,+D;MAsBA,gD;MatBA,uC;QAMW,kBAAU,gB;QAoBD,Q;QAAA,OjBqRoE,iBAAQ,W;QiBrR5F,OAAgB,cAA  
hB,C;UAAgB,yB;UACZ,WArB6B,SAqBIB,CAAU,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAtBhB,OA  
wBO,W;O;KA9BX,C;uFASA,yB;MAAA,+D;MAwBA,gD;MAxBA,uC;QAUW,kBAAU,gB;QAsBD,Q;QAAA,Oj  
BsQoE,iBAAQ,W;QiBtQ5F,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAvB6B,SAuBIB,CAAU,OAAV,C;UACC,O  
AAZ,WAAY,EAAO,IAAP,C;;QAxBhB,OA0BO,W;O;KApCX,C;2FAaA,yB;MAAA,gD;MAAA,oD;QAIoB,Q;QA  
AA,OAAA,SjBqRoE,QAAQ,W;QiBrR5F,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAW,UAAU,OAAV,C;UACC  
,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KARX,C;2FAWA,yB;MAAA,gD;MAAA,oD;QAQoB,Q;Q  
AAA,OAAA,SjBsQoE,QAAQ,W;QiBtQ5F,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAW,UAAU,OAAV,C;UAC  
C,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KAZX,C;8EAeA,yB;MAAA,gE;MAAA,uC;QAOW,kBA  
AM,eAAa,cAAb,C;QA2BA,Q;QAAA,OjB6NuE,iBAAQ,W;QiB7N5F,OAAa,cAAb,C;UAAa,sB;UACT,WAAY,W  
A5BiB,SA4Bb,CAAU,IAAV,CAAJ,C;;QA5BhB,OA6BO,W;O;KApCX,C;4FAUA,yB;MAAA,+D;MAAA,uC;QA  
OW,kBAaA,gB;QA4EJ,Q;QAAA,OjBkKoE,iBAAQ,W;QiBIK5F,OAAgB,cAAhB,C;UAAgB,yB;UApEK,U;UAA  
A,cARe,SAQf,CAoEQ,OApER,W;YAAsC,6B;;QAR3D,OASO,W;O;KAhBX,C;gGAUA,yB;MAAA,oD;QAyEoB,  
Q;QAAA,OjBkKoE,iBAAQ,W;QiBIK5F,OAAgB,cAAhB,C;UAAgB,yB;UApEK,U;UAAA,wBAoEQ,OApER,W;  
YAAsC,6B;;QAC3D,OAAO,W;O;KANX,C;kFASA,6C;MAKiB,Q;MAAA,OAAA,SjB6NuE,QAAQ,W;MiB7N5F,  
OAAa,cAAb,C;QAAa,sB;QACT,WAAY,WAAI,UAAU,IAAV,CAAJ,C;;MAChB,OAAO,W;K;8EAGX,gC;MAOo  
B,Q;MADhB,IAAI,mBAAJ,C;QAae,OAAO,I;MACN,OAAA,SjBiNoE,QAAQ,W;MiBjN5F,OAAgB,cAAhB,C;Q  
AAgB,yB;QAAM,IAAI,CAAC,UAAU,OAAV,CAAL,C;UAAyB,OAAO,K;;MACtD,OAAO,I;K;IAGX,2B;MAMI,  
OAAO,CAAC,mB;K;+EAGZ,gC;MAOoB,Q;MADhB,IAAI,mBAAJ,C;QAae,OAAO,K;MACN,OAAA,SjB6LoE,  
QAAQ,W;MiB7L5F,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,I;;MA  
CrD,OAAO,K;K;mFAGX,qB;MAKI,OAAO,c;K;mFAGX,gC;MAMoB,Q;MAFhB,IAAI,mBAAJ,C;QAae,OAAO,  
C;MACtB,YAAY,C;MACI,OAAA,SjB2KoE,QAAQ,W;MiB3K5F,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,U  
AAU,OAAV,CAAJ,C;UAAwB,qB;;MAC9C,OAAO,K;K;sFAGX,6B;MAKoB,Q;MAAA,OAAA,SjBkKoE,QAAQ  
,W;MiBIK5F,OAAgB,cAAhB,C;QAAgB,yB;QAAM,OAAO,OAAP,C;;K;kFAG1B,+B;MAemB,kBAAR,iB;MAA  
Q,sB;;QJkoDf,eAAe,sB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,I;UAAP,uB;SACzB,cAAc,QAAS,O  
;QACvB,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,O;UAAP,uB;SACzB,eIjpDmB,QJipDJ,CAAS,OAAT,C;;  
UAEX,QAAQ,QAAS,O;UACjB,QIppDe,QJopDP,CAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,  
UAAU,C;YACV,WAAW,C;;QAED,QAAT,QAAS,W;QACIB,qBAAO,O;;MI1pDP,yB;K;8FAGJ,+B;MAQmB,kB  
AAR,iB;MAAQ,sB;;QJkoDf,eAAe,sB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,I;UAAP,uB;SACzB,c  
AAc,QAAS,O;QACvB,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,O;UAAP,uB;SACzB,eItoD2B,QJsoDZ,CA  
AS,OAAT,C;;UAEX,QAAQ,QAAS,O;UACjB,QIzoDuB,QJyoDf,CAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,K  
AAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAED,QAAT,QAAS,W;QACIB,qBAAO,O;;MI/oDP,yB;K;mFAGJ,y  
B;MJ+oDA,sE;MF/yDA,iB;MMgKA,sC;QJ4pDI,eI/oDO,iBJ+oDQ,W;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAy  
B,MAAM,6B;QAC/B,eIjpDqB,QJipDN,CAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QInpDiB,  
QJmpDT,CAAS,QAAS,OAAIB,C;UACR,WFzzDG,MAAO,KEyzDO,QFzzDP,EEyzDiB,CFzzDjB,C;;QMqKd,OJs

pDO,Q;O;KInqDX,C;mFAGBA,yB;MJspDA,sE;MFj1DA,iB;MM2LA,sC;QJmqDI,eItpDO,iBJspDQ,W;QACf,IAA  
I,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;QAC/B,eIxpDqB,QJwpDN,CAAS,QAAS,OAAIB,C;QACf,OAAO,  
QAAS,UAAhB,C;UACI,QI1pDiB,QJ0pDT,CAAS,QAAS,OAAIB,C;UACR,WF31DG,MAAO,KE21DO,QF31DP,  
EE21DiB,CF31DjB,C;;QMgMd,OJ6pDO,Q;O;K11qDX,C;mFAGBA,yB;MJ6pDA,sE;MI7pDA,sC;QJwqDI,eI7pDO  
,iBJ6pDQ,W;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;QAC/B,eI/pDqB,QJ+pDN,CAAS,QAAS,O  
AAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QIjqDiB,QJiqDT,CAAS,QAAS,OAAIB,C;UACR,IAAI,2BAAW,  
CAAX,KAJ,C;YACI,WAAW,C;;QInqDnB,OJsqDO,Q;O;KIjrDX,C;+FACa,yB;MN9MA,iB;MM8MA,sC;QAW  
mB,kBAAR,iB;QAAQ,sB;;UJsqDf,eAAe,sB;UACf,IAAI,CAAC,QAAS,UAAAd,C;YAAyB,qBAAO,I;YAAP,uB;W  
ACzB,eIxd2B,QJwqDZ,CAAS,QAAS,OAAIB,C;UACf,OAAO,QAAS,UAAhB,C;YACI,QI1qDuB,QJ0qDf,CAA  
S,QAAS,OAAIB,C;YACR,WF53DG,MAAO,KE43DO,QF53DP,EE43DiB,CF53DjB,C;;UE83Dd,qBAAO,Q;;QI7  
qDP,yB;O;KAXJ,C;+FACa,yB;MNvOA,iB;MMuOA,sC;QAWmB,kBAAR,iB;QAAQ,sB;;UJ6qDf,eAAe,sB;UACf  
,IAAI,CAAC,QAAS,UAAAd,C;YAAyB,qBAAO,I;YAAP,uB;WACzB,eI/qD2B,QJ+qDZ,CAAS,QAAS,OAAIB,C;U  
ACf,OAAO,QAAS,UAAhB,C;YACI,QIjrDuB,QJirDf,CAAS,QAAS,OAAIB,C;YACR,WF55DG,MAAO,KE45DO  
,QF55DP,EE45DiB,CF55DjB,C;;UIprDP,yB;O;KAXJ,C;+FACa,+B;MASmB,kBAAR,iB;M  
AAQ,sB;;QJorDf,eAAe,sB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,I;YAAP,uB;SACzB,eItrD2B,QJ  
rDZ,CAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QIxrDuB,QJwrDf,CAAS,QAAS,OAAIB,C;U  
ACR,IAAI,2BAAW,CAAX,KAJ,C;YACI,WAAW,C;;QAGnB,qBAAO,Q;;MI7rDP,yB;K;0FAGJ,yB;MJ6rDA,s  
E;MI7rDA,kD;QJwsDI,eI7rDO,iBJ6rDQ,W;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;QAC/B,eI/r  
DqC,QJ+rDtB,CAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QIjsDiC,QJisDzB,CAAS,QAAS,O  
AAIB,C;UACR,IIIsDqB,UJksDN,SAAQ,QAAR,EAakB,CAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QIn  
sDnB,OJssDO,Q;O;KIjtDX,C;sGAcA,2C;MASmB,kBAAR,iB;MAAQ,0B;;QJssDf,eAAe,sB;QACf,IAAI,CAAC,Q  
AAS,UAAAd,C;UAAyB,yBAAO,I;YAAP,2B;SACzB,eIxsD2C,QJwsD5B,CAAS,QAAS,OAAIB,C;QACf,OAAO,Q  
AAS,UAAhB,C;UACI,QI1sDuC,QJ0sD/B,CAAS,QAAS,OAAIB,C;UACR,II3sD2B,UJ2sDZ,SAAQ,QAAR,EAak  
B,CAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,yBAAO,Q;;MI/sDP,6B;K;sFAGJ,yB;MAOA,8D;  
MAPA,wC;QAIL,OASe,cAAR,iBAAQ,EATM,UASN,C;O;KAbnB,C;kGAOA,yB;MAAA,8D;MAAA,wC;QAMI,  
OAAe,cAAR,iBAAQ,EAac,UAAAd,C;O;KANnB,C;kFASA,+B;MAcmB,kBAAR,iB;MAAQ,sB;;QJwxDf,eAAe,sB  
;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,I;YAAP,uB;SACzB,cAAc,QAAS,O;QACvB,IAAI,CAAC,  
QAAS,UAAAd,C;UAAyB,qBAAO,O;YAAP,uB;SACzB,eIvyDmB,QJuyDJ,CAAS,OAAT,C;;UAEX,QAAQ,QAAS,  
O;UACjB,QI1yDe,QJ0yDP,CAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAJ,C;YACI,UAAU,C;YACV,WAA  
W,C;;QAED,QAAT,QAAS,W;QACIB,qBAAO,O;;MIhzDP,yB;K;8FAGJ,+B;MAQmB,kBAAR,iB;MAAQ,sB;;QJ  
wxDf,eAAe,sB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,I;YAAP,uB;SACzB,cAAc,QAAS,O;QACvB  
,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,O;YAAP,uB;SACzB,eI5xD2B,QJ4xDZ,CAAS,OAAT,C;;UAEX,  
QAAQ,QAAS,O;UACjB,QI/xDuB,QJ+xDf,CAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAJ,C;YACI,UAAU,  
C;YACV,WAAW,C;;QAED,QAAT,QAAS,W;QACIB,qBAAO,O;;MIryDP,yB;K;mFAGJ,yB;MJqyDA,sE;MFI4D  
A,iB;MM6FA,sC;QJkzDI,eIryDO,iBJqyDQ,W;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;QAC/B,eI  
vyDqB,QJuyDN,CAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QIzyDiB,QJyyDT,CAAS,QAAS,  
OAAIB,C;UACR,WF54DG,MAAO,KE44DO,QF54DP,EE44DiB,CF54DjB,C;;QMkGd,OJ4yDO,Q;O;KIzzDX,C;  
mFAGBA,yB;MJ4yDA,sE;MFp6DA,iB;MMwHA,sC;QJyzDI,eI5yDO,iBJ4yDQ,W;QACf,IAAI,CAAC,QAAS,UA  
Ad,C;UAAyB,MAAM,6B;QAC/B,eI9yDqB,QJ8yDN,CAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;U  
ACI,QIhzDiB,QJgzDT,CAAS,QAAS,OAAIB,C;UACR,WF96DG,MAAO,KE86DO,QF96DP,EE86DiB,CF96DjB,  
C;;QM6Hd,OJmzDO,Q;O;KIh0DX,C;mFAGBA,yB;MJmzDA,sE;MIInzDA,sC;QJ8zDI,eInzDO,iBJmzDQ,W;QACf,  
IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;QAC/B,eIrzDqB,QJqzDN,CAAS,QAAS,OAAIB,C;QACf,OAA  
O,QAAS,UAAhB,C;UACI,QIvzDiB,QJuzDT,CAAS,QAAS,OAAIB,C;UACR,IAAI,2BAAW,CAAX,KAJ,C;YA  
CI,WAAW,C;;QIzzDnB,OJ4zDO,Q;O;KIv0DX,C;+FACa,yB;MN3IA,iB;MM2IA,sC;QAWmB,kBAAR,iB;QAAQ  
,sB;;UJ4zDf,eAAe,sB;UACf,IAAI,CAAC,QAAS,UAAAd,C;YAAyB,qBAAO,I;YAAP,uB;WACzB,eI9zD2B,QJ8zD  
Z,CAAS,QAAS,OAAIB,C;UACf,OAAO,QAAS,UAAhB,C;YACI,QIh0DuB,QJg0Df,CAAS,QAAS,OAAIB,C;YA  
CR,WF/8DG,MAAO,KE+8DO,QF/8DP,EE+8DiB,CF/8DjB,C;;UEi9Dd,qBAAO,Q;;QIn0DP,yB;O;KAXJ,C;+FAC  
A,yB;MNpKA,iB;MMoKA,sC;QAWmB,kBAAR,iB;QAAQ,sB;;UJm0Df,eAAe,sB;UACf,IAAI,CAAC,QAAS,UA

Ad,C;YAAyB,qBAAO,I;YAAP,uB;WACzB,eIr0D2B,QJq0DZ,CAAS,QAAS,OAAIB,C;UACf,OAAO,QAAS,UA  
AhB,C;YACI,QIv0DuB,QJu0Df,CAAS,QAAS,OAAIB,C;YACR,WF/+DG,MAAO,KE++DO,QF/+DP,EE++DiB,C  
F/+DjB,C;;UEi/Dd,qBAAO,Q;;;QI10DP,yB;O;KAXJ,C;+FACa,+B;MASmB,kBAAR,iB;MAAQ,sB;;QJ00Df,eAA  
e,sB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,I;UAAP,uB;SACzB,eI50D2B,QJ40DZ,CAAS,QAAS,O  
AAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QI90DuB,QJ80Df,CAAS,QAAS,OAAIB,C;UACR,IAAI,2BAAW,  
CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,qBAAO,Q;;;MIn1DP,yB;K;0FAGJ,yB;MJm1DA,sE;MIn1DA,kD;QJ8  
1DI,eIn1DO,iBjM1DQ,W;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;QAC/B,eIr1DqC,QJq1DtB,CA  
AS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QIv1DiC,QJu1DzB,CAAS,QAAS,OAAIB,C;UACR,I  
Ix1DqB,UJw1DN,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QIz1DnB,OJ41DO,  
Q;O;KIv2DX,C;sGAcA,2C;MASmB,kBAAR,iB;MAAQ,0B;;QJ41Df,eAAe,sB;QACf,IAAI,CAAC,QAAS,UAAAd,  
C;UAAyB,yBAAO,I;UAAP,2B;SACzB,eI91D2C,QJ81D5B,CAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAh  
B,C;UACI,QIh2DuC,QJg2D/B,CAAS,QAAS,OAAIB,C;UACR,Ij2D2B,UJi2DZ,SAAQ,QAAR,EAakB,CAAIB,C  
AAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,yBAAO,Q;;;Mlr2DP,6B;K;IAGJ,0C;MAGI,OASe,gBAAR,iB  
AAQ,EATM,UASN,C;K;kGANnB,yB;MAAA,8D;MAAA,wC;QAMI,OAAe,cAAR,iBAAQ,EAAC,UAAAd,C;O;KA  
NnB,C;IASA,4B;MAMI,OAAO,mB;K;iFAGX,gC;MAOoB,Q;MADhB,IAAI,mBAAJ,C;QAae,OAAO,I;MACN,O  
AAA,SjBnJoE,QAAQ,W;MiBmJ5F,OAAgB,cAAhB,C;QAAGB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAw  
B,OAAO,K;;MACrD,OAAO,I;K;oFAGX,6B;MAKmC,Q;MAAA,OjB5JqD,iBAAQ,W;MiB4J7E,OAAgB,cAAhB,  
C;QAAGB,yB;QAAM,OAAO,OAAP,C;;MAArC,gB;K;kGAGJ,yB;MAAA,6B;MAAA,sC;MJwyCA,wE;MIxyCA,  
2BAQiB,yB;QJgyCjB,wE;eIhyCiB,0B;UAAA,4B;YAAU,kBAAR,iB;YAAQ,aAAe,c;YJuyCzB,gB;YADb,YAAY,  
C;YACC,6B;YAAb,OAAa,cAAb,C;cAAa,sB;cAAM,OAAO,oBAAmB,cAAnB,EAAMB,sBAAnB,UAAP,EAaoC,  
IAApC,C;;YIvyC2B,W;W;S;OAAjC,C;MARjB,oC;QJ+yCiB,gB;QADb,YAAY,C;QACC,OIvyCE,iBjuyCF,W;QA  
Ab,OAAa,cAAb,C;UAAa,sB;UAAM,OAAO,oBAAmB,cAAnB,EAAMB,sBAAnB,UAAP,EAaoC,IAApC,C;;QIvy  
CnB,gB;O;KARJ,C;4FAWA,qB;MAKI,OAAO,iB;K;IAGX,iC;MAIL,OAAe,aAAR,iBAAQ,C;K;IC9hBnB,kC;MA  
EI,gBCmE2D,8BAAy,c;MDIEvE,IAAI,SAAU,OAAV,GAAMB,CAAvB,C;QACW,Q;QAAA,IAAI,cAAQ,GAZ,  
C;UAAA,OAAsB,S;;uBAae,qBAAU,CAAV,C;UAAA,YAAe,SEiNc,WFjNM,CEiNN,CAff,c;UFIMnD,OG8MoD,  
2BAAL,GAakB,K;;QH9MxE,W;OAEJ,OAAuB,oBAAhB,wBAAGB,C;K;gFxBd3B,yB;MAAA,mC;MAAA,2C;M  
AAA,4B;QAQI,OAAO,kBAAO,cAAP,C;O;KARX,C;gFAWA,yB;MAAA,mC;MAAA,2C;MAAA,4B;QAQI,OAA  
O,kBAAO,cAAP,C;O;KARX,C;gFAWA,yB;MAAA,mC;MAAA,2C;MAAA,4B;QAQI,OAAO,kBAAO,cAAP,C;O  
;KARX,C;IAWA,sC;;QAQQ,OAAc,QAAP,MAAO,EAQ,SAAR,C;;QACHB,+C;UACE,MAAM,2BAAuB,CAAE  
,QAAzB,C;;UAHV,O;;K;IAOJ,sC;;QAQQ,OAAc,SAAP,MAAO,EAAS,SAAT,C;;QACHB,+C;UACE,MAAM,2B  
AAuB,CAAE,QAAzB,C;;UAHV,O;;K;IAOJ,sC;;QAQQ,OAAiD,OAA1C,MAAO,iBAAQ,e4BtCgB,I5BsCxB,EA  
AoB,CAAA,c4BtCl,I5BsCJ,IAAY,CAAZ,IAApB,CAAmC,C;;QACnD,+C;UACE,MAAM,2BAAuB,CAAE,QAAz  
B,C;;UAHV,O;;K;4FAOJ,yB;MAAA,mC;MAAA,uD;MAAA,4B;QAOI,OAAO,wBAAa,cAAb,C;O;KAPX,C;4FA  
UA,yB;MAAA,mC;MAAA,uD;MAAA,4B;QAOI,OAAO,wBAAa,cAAb,C;O;KAPX,C;4FAUA,yB;MAAA,mC;M  
AAA,uD;MAAA,4B;QAOI,OAAO,wBAAa,cAAb,C;O;KAPX,C;IAUA,4C;MAMI,IAAI,mBAAJ,C;QACI,OAAO,  
I;MACX,OAAc,QAAP,MAAO,EAQ,SAAR,C;K;IAGIB,4C;MAMI,IAAI,mBAAJ,C;QACI,OAAO,I;MACX,OA  
Ac,SAAP,MAAO,EAAS,SAAT,C;K;IAGIB,4C;MAMI,IAAI,mBAAJ,C;QACI,OAAO,I;MACX,OAAiD,OAA1C,  
MAAO,iBAAQ,e4BxGoB,I5BwG5B,EAaoB,CAAA,c4BxGQ,I5BwGR,IAAY,CAAZ,IAApB,CAAmC,C;K;mFA  
GrD,8B;MAQI,OAAO,mBAAmB,2BAAS,OAAT,C;K;oFAG9B,8B;MAQI,OAAO,mBAAmB,2BAAS,OAAT,C;K  
;oFAG9B,8B;MAQI,OAAO,mBAAmB,2BAAS,OAAT,C;K;IAG9B,uC;MAKI,OAAO,2BAae,KAAf,C;K;IAGX,u  
C;MAKI,OAAO,2BAae,oBAAN,KAAM,CAaf,C;K;IAGX,uC;MAKI,OAAO,2BAae,KAAf,C;K;IAGX,uC;MAO  
I,OAAO,2BAae,KAAf,C;K;IAGX,uC;MAOI,OAAO,2BAae,KAAf,C;K;IAGX,uC;MgBzHW,ShBgIM,mBAAN,  
KAAM,C;MAAb,OAA0C,UAAJ,GAAgB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG5E,uC;MgBnIW,ShB0IM,kB  
AAN,KAAM,C;MAAb,OAA2C,UAAJ,GAAgB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG7E,uC;MgB7IW,ShBoJ  
M,oBAAN,KAAM,C;MAAb,OAA2C,UAAJ,GAAgB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG7E,uC;MgBvJW,S  
hB8JM,qBAAN,KAAM,C;MAAb,OAA4C,UAAJ,GAAgB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG9E,uC;MAKI  
,OAAO,2BAae,KAAf,C;K;IAGX,uC;MgBzKW,ShBgLM,mBAAN,KAAM,C;MAAb,OAA0C,UAAJ,GAAgB,2B  
AAS,EAAT,CAAhB,GAakC,K;K;IAG5E,uC;MgBnLW,ShB0LM,oBAAN,KAAM,C;MAAb,OAA2C,UAAJ,GAA

gB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG7E,uC;MgB7LW,ShBoMM,oBAAN,KAAM,C;MAAb,OAA2C,UAA  
J,GAAgB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG7E,uC;MgBvMW,ShB8MM,qBAAN,KAAM,C;MAAb,OAA4  
C,UAAJ,GAAgB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG9E,uC;MAKI,OAAO,2BA Ae,KAAf,C;K;IAGX,uC;M  
AKI,OAAO,2BA Ae,oBAAN,KAAM,CAAf,C;K;IAGX,uC;MgBjOW,ShBsOM,kBAAN,KAAM,C;MAAb,OAA2C  
,UAAJ,GAAgB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG7E,uC;MgBzOW,ShB8OM,mBAAN,KAAM,C;MAAb,  
OAA4C,UAAJ,GAAgB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG9E,uC;MAOI,OAAO,2BA Ae,KAAf,C;K;IAGX  
,uC;MAOI,OAAO,2BA Ae,KAAf,C;K;IAGX,uC;MgBrQW,ShB0QM,iBAAN,KAAM,C;MAAb,OAA0C,UAAJ,G  
AAgB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG5E,uC;MgB7QW,ShBkRM,oBAAN,KAAM,C;MAAb,OAA2C,U  
AAJ,GAAgB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG7E,uC;MgBrRW,ShB0RM,qBAAN,KAAM,C;MAAb,OA  
A4C,UAAJ,GAAgB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG9E,uC;MAOI,OAAO,2BAAS,KAAM,WAAf,C;K;I  
AGX,uC;MAOI,OAAO,2BAAS,KAAM,WAAf,C;K;IAGX,uC;MAKI,OAAO,2BA Ae,KAAf,C;K;IAGX,uC;MAKI  
,OAAO,2BA Ae,oBAAN,KAAM,CAAf,C;K;IAGX,uC;MgBjUW,ShBsUM,oBAAN,KAAM,C;MAAb,OAA2C,UA  
AJ,GAAgB,2BAAS,EAAT,CAAhB,GAakC,K;K;IAG7E,uC;MAOI,OAAO,2BA Ae,KAAf,C;K;IAGX,uC;MAOI,  
OAAO,2BA Ae,KAAf,C;K;IAGX,+B;MAOI,OAAO,sCA Ae,yBAAgB,SAAhB,EAAyB,EAazB,EAakC,EAaIC,C;  
K;IAG1B,iC;MAOI,OAAO,uCAAgB,yBAAgB,SAAhB,EAAyB,oBAAH,EAAG,CAAzB,M;K;IAG3B,iC;MAOI,O  
AAO,sCA Ae,yBAAgB,SAArB,EAAiC,EAAjC,EAA0C,EAA1C,C;K;IAG1B,iC;MAOI,OAAO,sCA Ae,yBAAgB,S  
AArB,EAAiC,EAAjC,EAA0C,EAA1C,C;K;IAG1B,iC;MAOI,OAAO,uCAAgB,yBAAgB,SAAhB,EAAyB,EAaB,  
EAA0B,EAA1B,C;K;IAG3B,iC;MAOI,OAAO,sCA Ae,yBAAgB,SAAhB,EAAyB,EAaB,EAA0B,EAA1B,C;K;IA  
G1B,iC;MAOI,OAAO,uCAAgB,yBAAgB,SAAhB,EAAyB,oBAAH,EAAG,CAAzB,M;K;IAG3B,iC;MAOI,OAA  
O,sCA Ae,yBAAgB,SAArB,EAA8B,EAA9B,EAakC,EAaIC,C;K;IAG1B,iC;MAOI,OAAO,sCA Ae,yBAAgB,SA  
ArB,EAA8B,EAA9B,EAakC,EAaIC,C;K;IAG1B,iC;MAOI,OAAO,uCAAgB,yBAAgB,oBAAL,SAAK,CAArB,EA  
A+B,EAA/B,M;K;IAG3B,iC;MAOI,OAAO,uCAAgB,yBAAgB,SAAhB,EAAyB,EAaB,M;K;IAG3B,kC;MAOI,O  
AAO,uCAAgB,yBAAgB,oBAAL,SAAK,CAArB,EAA+B,EAA/B,M;K;IAG3B,kC;MAOI,OAAO,uCAAgB,yBA  
AgB,oBAAL,SAAK,CAArB,EAA+B,EAA/B,M;K;IAG3B,kC;MAOI,OAAO,sCA Ae,yBAAgB,SAAhB,EAAyB,EA  
AzB,EAakC,EAaIC,C;K;IAG1B,kC;MAOI,OAAO,uCAAgB,yBAAgB,SAAhB,EAAyB,oBAAH,EAAG,CAAzB,  
M;K;IAG3B,kC;MAOI,OAAO,sCA Ae,yBAAgB,SAArB,EAAiC,EAAjC,EAA0C,EAA1C,C;K;IAG1B,kC;MAOI,  
OAAO,sCA Ae,yBAAgB,SAArB,EAAiC,EAAjC,EAA0C,EAA1C,C;K;IAG1B,+B;MAII,OAAO,sCA Ae,yBAAgB,  
cAAhB,EAAyB,EAaB,EAA6B,CAAC,cAAD,IAA7B,C;K;IAG1B,gC;MAII,OAAO,uCAAgB,yBAAgB,cAAhB,E  
AAyB,EAaB,EAA8B,cAAD,aAA7B,C;K;IAG3B,gC;MAII,OAAO,uCAAgB,yBAAgB,cAAhB,EAAyB,EAaB,E  
AA6B,CAAC,cAAD,IAA7B,C;K;IAG3B,+B;MAII,oBAAoB,OAAO,CAA3B,EAA8B,IAA9B,C;MACA,OAAO,s  
CA Ae,yBAAgB,eAAhB,EAaB,cAAvB,EAAiC,SAAK,KAAL,GAAY,CAAhB,GAAMB,IAANB,GAA6B,CAAC,  
IAAD,IAA1D,C;K;IAG1B,iC;MAII,oBAAoB,kBAAO,CAA3B,EAA8B,IAA9B,C;MACA,OAAO,uCAAgB,yBA  
AgB,eAAhB,EAaB,cAAvB,EAAiC,SAAK,KAAL,cAAy,CAAhB,GAAMB,IAANB,GAA6B,CAAC,IAAD,IAA1D,C;K;I  
AG3B,iC;MAII,oBAAoB,OAAO,CAA3B,EAA8B,IAA9B,C;MACA,OAAO,uCAAgB,yBAAgB,eAAhB,EAaB,c  
AAvB,EAAiC,SAAK,KAAL,GAAY,CAAhB,GAAMB,IAANB,GAA6B,CAAC,IAAD,IAA1D,C;K;IAG3B,sC;MA  
CI,OAAmB,IAAR,8BAAGC,GAApC,GAAiE,OAAL,SAAK,CAAjE,GAA+E,I;K;IAG1F,wC;MACI,OAAW,mEA  
AJ,GAAMe,OAAL,SAAK,SAAnE,GAAiF,I;K;IAG5F,wC;MACI,OAAW,YAAQ,aAAA,sCA Ae,UAAf,EAA0B,s  
CA Ae,UAAzC,CAAR,YAAJ,GAAqE,OAAL,SAAK,CAArE,GAAMF,I;K;IAG9F,wC;MACI,OAAmB,UAAA,sCA  
Ae,UAAf,EAA2B,sCA Ae,UAA1C,CAAR,4BAAJ,GAA+E,OAAR,YAAL,SAAK,CAAQ,CAA/E,GAA6F,I;K;IAG  
xG,wC;MACI,OAAmB,UAAA,sCA Ae,UAAf,EAA0B,sCA Ae,UAAzC,CAAR,4BAAJ,GAA6E,OAAR,YAAL,SA  
AK,CAAQ,CAA7E,GAA2F,I;K;IAGtG,qC;MACI,OAAW,iFAAJ,GAA4D,SAAK,QAAjE,GAA8E,I;K;IAGzF,uC;  
MACI,OAAmB,UAAc,WAAAd,EAAwC,UAAxC,CAAR,4BAAJ,GAAqE,YAAL,SAAK,CAArE,GAakF,I;K;IAG7  
F,uC;MACI,OAAmB,UAAc,WAAAd,EAAuC,UAAvC,CAAR,4BAAJ,GAAmE,YAAL,SAAK,CAAnE,GAAGf,I;K;  
IAG3F,sC;MACI,OAAmB,UAAe,mCAAf,EAA0C,mCAA1C,CAAR,4BAAJ,GAAuE,uBAAL,SAAK,CAAvE,GA  
AqF,I;K;IAGhG,wC;MACI,OAAmB,UAAe,mCAAf,EAAyC,mCAAzC,CAAR,4BAAJ,GAAqE,uBAAL,SAAK,C  
AArE,GAAMF,I;K;IAG9F,uC;MACI,OAAmB,MAAR,8BAAiC,KAAR,C,GAAMe,QAAL,SAAK,CAAnE,GAakF,  
I;K;IAG7F,yC;MACI,OAAW,uEAAJ,GAAqE,QAAL,SAAK,SAArE,GAAoF,I;K;IAG/F,yC;MACI,OAAmB,UAA  
A,uCAAgB,UAAhB,EAA4B,uCAAgB,UAA5C,CAAR,4BAAJ,GAAiF,QAAR,YAAL,SAAK,CAAQ,CAAjF,GAA

gG,I;K;IAG3G,yC;MACI,OOAmB,UAAA,uCAAgB,UAAhB,EAA2B,uCAAgB,UAA3C,CAAR,4BAAJ,GAA+E,QAAR,YAAL,SAAK,CAAQ,CAA/E,GAA8F,I;K;IAGzG,8B;MAMI,OOAO,wBAAY,EAAa,GAAH,CAAG,IAAzB,C;K;IAGX,gC;MAMI,OOAO,kBAAY,oBAAH,EAAG,CAAc,8BAAH,CAAG,EAA1B,C;K;IAGX,gC;MAMI,OOAO,aAAK,SAAL,EAAoB,EAAa,GAAH,CAAG,IAAjC,C;K;IAGX,gC;MAMI,OOAO,aAAK,SAAL,EAAoB,EA Aa,GAAH,CAAG,IAAjC,C;K;IAGX,gC;MAMI,IAAI,MAAM,CAAV,C;QAAoB,OOAO,iCAAU,M;MACrC,OOAO,yBAAiB,OOAR,EAAQ,GAAH,CAAG,CAAjB,C;K;IAGX,gC;MAMI,IAAI,MAAM,WAAV,C;QAAyB,OOAO,gCAAS,M;MACzC,OOAO,wBAAS,EAAQ,GAAH,CAAG,IAAjB,C;K;IAGX,gC;MAMI,OOAO,kBAAY,oBAAH,EAAG,CAAc,8BAAH,CAAG,EAA1B,C;K;IAGX,gC;MAMI,IAAI,MAAM,WAAV,C;QAAyB,OOAO,gCAAS,M;MACzC,OOAO,aAAK,SAAL,EAAiB,EAAQ,GAAH,CAAG,IAAzB,C;K;IAGX,gC;MAMI,IAAI,MAAM,WAAV,C;QAAyB,OOAO,gCAAS,M;MACzC,OOAO,aAAK,SAAL,EAAiB,EAAQ,GAAH,CAAG,IAAzB,C;K;IAGX,gC;MAMI,IAAI,iDAAJ,C;QAA0B,OOAO,iCAAU,M;MAC3C,OOAY,oBAAL,SAAK,CAAL,SAAkB,EAAQ,8BAAH,CAAG,EAA1B,C;K;IAGX,gC;MAMI,IAAI,iDAAJ,C;QAA0B,OOAO,iCAAU,M;MAC3C,OOAO,kBAAS,EAAQ,8BAAH,CAAG,EAAjB,C;K;IAGX,iC;MAMI,IAAI,iDAAJ,C;QAA0B,OOAO,iCAAU,M;MAC3C,OOAY,oBAA L,SAAK,CAAL,SAAkB,EAAQ,8BAAH,CAAG,EAA1B,C;K;IAGX,iC;MAMI,IAAI,iDAAJ,C;QAA0B,OOAO,iC AAU,M;MAC3C,OOAY,oBAAL,SAAK,CAAL,SAAkB,EAAQ,8BAAH,CAAG,EAA1B,C;K;IAGX,iC;MAMI,OO AAO,wBAAY,EAAa,GAAH,CAAG,IAAzB,C;K;IAGX,iC;MAMI,OOAO,kBAAY,oBAAH,EAAG,CAAc,8BAAH,CAAG,EAA1B,C;K;IAGX,iC;MAMI,OOAO,aAAK,SAAL,EAAoB,EAAa,GAAH,CAAG,IAAjC,C;K;IAGX,iC;MAMI,OOAO,aAAK,SAAL,EAAoB,EAAa,GAAH,CAAG,IAAjC,C;K;IAGX,gD;MAQI,OOAW,4BAAO,YAAP, KAAJ,GAAyB,YAAzB,GAA2C,S;K;IAGtD,kD;MAQI,OOAW,YAAO,YAAX,GAAyB,YAAzB,GAA2C,S;K;IAG tD,kD;MAQI,OOAW,YAAO,YAAX,GAAyB,YAAzB,GAA2C,S;K;IAGtD,kD;MAQI,OOAW,YAAO,YAAX,GA AyB,YAAzB,GAA2C,S;K;IAGtD,kD;MAQI,OOAW,0BAAO,YAAP,KAAJ,GAAyB,YAAzB,GAA2C,S;K;IAGtD ,kD;MAQI,OOAW,YAAO,YAAX,GAAyB,YAAzB,GAA2C,S;K;IAGtD,kD;MAQI,OOAW,YAAO,YAAX,GAAy B,YAAzB,GAA2C,S;K;IAGtD,+C;MAQI,OOAW,4BAAO,YAAP,KAAJ,GAAyB,YAAzB,GAA2C,S;K;IAGtD,iD ;MAQI,OOAW,YAAO,YAAX,GAAyB,YAAzB,GAA2C,S;K;IAGtD,iD;MAQI,OOAW,YAAO,YAAX,GAAyB,Y AAzB,GAA2C,S;K;IAGtD,iD;MAQI,OOAW,YAAO,YAAX,GAAyB,YAAzB,GAA2C,S;K;IAGtD,iD;MAQI,OA AW,0BAAO,YAAP,KAAJ,GAAyB,YAAzB,GAA2C,S;K;IAGtD,iD;MAQI,OOAW,YAAO,YAAX,GAAyB,YAA zB,GAA2C,S;K;IAGtD,iD;MAQI,OOAW,YAAO,YAAX,GAAyB,YAAzB,GAA2C,S;K;IAGtD,yD;MAQI,IAAI,i BAAiB,IAAjB,IAAyB,iBAAiB,IAA9C,C;QACI,IAAI,+BAAe,YAAf,KAAJ,C;UAAiC,MAAM,gCAAyB,6DAAiD ,YAAjD,wCAAoF,YAApF,OOAzB,C;QACvC,IAAI,4BAAO,YAAP,KAAJ,C;UAAyB,OOAO,Y;QAChC,IAAI,4B AAO,YAAP,KAAJ,C;UAAyB,OOAO,Y;;QAGhC,IAAI,iBAAiB,IAAjB,IAAyB,4BAAO,YAAP,KAA7B,C;UAAk D,OOAO,Y;QACzD,IAAI,iBAAiB,IAAjB,IAAyB,4BAAO,YAAP,KAA7B,C;UAAkD,OOAO,Y;;MAE7D,OOAO, S;K;IAGX,2D;MAQI,IAAI,eAAe,YAAnB,C;QAAiC,MAAM,gCAAyB,oDAAiD,YAAjD,8BAAoF,YAApF,MAA zB,C;MACvC,IAAI,YAAO,YAAX,C;QAAyB,OOAO,Y;MACHC,IAAI,YAAO,YAAX,C;QAAyB,OOAO,Y;MAC hC,OOAO,S;K;IAGX,2D;MAQI,IAAI,eAAe,YAAnB,C;QAAiC,MAAM,gCAAyB,oDAAiD,YAAjD,8BAAoF,YA ApF,MAAzB,C;MACvC,IAAI,YAAO,YAAX,C;QAAyB,OOAO,Y;MACHC,IAAI,YAAO,YAAX,C;QAAyB,OOA O,Y;MACHC,OOAO,S;K;IAGX,2D;MAQI,IAAI,eAAe,YAAnB,C;QAAiC,MAAM,gCAAyB,oDAAiD,YAAjD,8B AAoF,YAApF,MAAzB,C;MACvC,IAAI,YAAO,YAAX,C;QAAyB,OOAO,Y;MACHC,IAAI,YAAO,YAAX,C;QA AyB,OOAO,Y;MACHC,OOAO,S;K;IAGX,2D;MAQI,IAAI,6BAAe,YAAf,KAAJ,C;QAAiC,MAAM,gCAAyB,oD AAiD,YAAjD,yCAAoF,YAApF,iBAAzB,C;MACvC,IAAI,0BAAO,YAAP,KAAJ,C;QAAyB,OOAO,Y;MACHC,I AAI,0BAAO,YAAP,KAAJ,C;QAAyB,OOAO,Y;MACHC,OOAO,S;K;IAGX,2D;MAQI,IAAI,eAAe,YAAnB,C;QA AiC,MAAM,gCAAyB,oDAAiD,YAAjD,8BAAoF,YAApF,MAAzB,C;MACvC,IAAI,YAAO,YAAX,C;QAAyB,OO AAO,Y;MACHC,IAAI,YAAO,YAAX,C;QAAyB,OOAO,Y;MACHC,OOAO,S;K;IAGX,2D;MAQI,IAAI,eAAe,YA AnB,C;QAAiC,MAAM,gCAAyB,oDAAiD,YAAjD,8BAAoF,YAApF,MAAzB,C;MACvC,IAAI,YAAO,YAAX,C; QAAyB,OOAO,Y;MACHC,IAAI,YAAO,YAAX,C;QAAyB,OOAO,Y;MACHC,OOAO,S;K;IAGX,sC;MAUW,Q; MADP,IAAI,KAAM,UAAV,C;QAAqB,MAAM,gCAAyB,4CAAyC,KAAzC,MAAzB,C;MAGvB,IAAA,KAAM,0 BAAiB,SAAjB,EAAuB,KAAM,MAA7B,CAAN,IAA6C,CAAC,KAAM,0BAAiB,KAAM,MAAvB,EAA8B,SAA9 B,CAApD,C;QAAiG,OOAN,KAAM,M;WAEjG,IAAA,KAAM,0BAAiB,KAAM,aAAvB,EAAqC,SAArC,CAAN,IA AoD,CAAC,KAAM,0BAAiB,SAAjB,EAAuB,KAAM,aAA7B,CAA3D,C;QAA+G,OOAN,KAAM,a;;QACvG,gB

;MALZ,W;K;IASJ,sC;MAYW,Q;MAJP,IAAI,8CAAJ,C;QACI,OAA Y,WAAL,SAAK,EAAY,KAAZ,C;OAEhB,IAAI,KAAM,UAAV,C;QAAqB,MAAM,gCAAYB,4CAAYC,KAAzC,MAAzB,C;MAEvB,gCAAO,KAAM,MAAb,M;QAA4B,OAAN,KAAM,M;WAC5B,gCAAO,KAAM,aAAb,M;QAAmC,OAAN,KAAM,a;QAC3B,gB;MAHZ,W;K;IAOJ,sC;MAYW,Q;MAJP,IAAI,8CAAJ,C;QACI,OAA Y,WAAL,SAAK,EAAC,KAA d,C;OAEhB,IAAI,KAAM,UAAV,C;QAAqB,MAAM,gCAAYB,4CAAYC,KAAzC,MAAzB,C;MAEvB,gBAAO,KAAM,MAAb,C;QAA4B,OAAN,KAAM,M;WAC5B,gBAAO,KAAM,aAAb,C;QAAmC,OAAN,KAAM,a;QAC3B,gB;MAHZ,W;K;IAOJ,sC;MAYW,Q;MAJP,IAAI,8CAAJ,C;QACI,OAA Y,WAAL,SAAK,EA Ae,KAAf,C;OAEhB,IAAI,KAAM,UAAV,C;QAAqB,MAAM,gCAAYB,4CAAYC,KAAzC,MAAzB,C;MAEvB,8BAAO,KAAM,MAAb,M;QAA4B,OAAN,KAAM,M;WAC5B,8BAAO,KAAM,aAAb,M;QAAmC,OAAN,KAAM,a;QAC3B,gB;MAHZ,W;K;IW1rCJ,oD;MAMuF,wC;K;IANvF,8CAOI,Y;MAAuC,8B;K;IAP3C,gF;IkBQA,yC;MAMI,OAAO,sBAAQ,OAAR,KAAoB,C;K;IAW G,2C;MAAA,qB;QAAE,MAAM,8BAA0B,+CAA4C,aaA5C,MAA1B,C;O;K;IAR1C,uC;MAQI,OAAO,8BAAgB,KAAhB,EAAuB,yBAAvB,C;K;IAGX,4D;MACqB,Q;MANjB,IAAI,QAAQ,CAAZ,C;QACI,OAAO,aAAa,KAAb,C;MACX,eAAe,oB;MACf,YAAY,C;MACZ,OAAO,QAAS,UAAhB,C;QACI,cAAc,QAAS,O;QACvB,IAAI,WAAS,YAAT,EAAS,oBAAT,OA AJ,C;UACI,OAAO,O;MAEf,OAAO,aAAa,KAAb,C;K;IAGX,8C;MACqB,Q;MANjB,IAAI,QAAQ,CAAZ,C;QACI,OAAO,I;MACX,eAAe,oB;MACf,YAAY,C;MACZ,OAAO,QAAS,UAAhB,C;QACI,cAAc,QAAS,O;QACvB,IAAI,WAAS,YAAT,EAAS,oBAAT,OA AJ,C;UACI,OAAO,O;MAEf,OAAO,I;K;8EAGX,gC;MASW,sB;QA2FS,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAGB,yB;UAAM,IA3FH,SA2FO,CAAU,OA AV,CAAJ,C;YAAwB,qBAAO,O;YAAP,uB;QAC9C,qBAAO,I;MA5FP,yB;K;uFAGJ,gC;MAkOoB,Q;MADhB,WAAe,I;MACC,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,IA1Nc,SA0NV,CAAU,OA AV,CAAJ,C;UACI,OAAO,O;MA3Nf,OA8NO,I;K;IA3NX,6B;MAOI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAA d,C;QACI,MAAM,2BAAuB,oBAAvB,C;MACV,OAAO,QAAS,O;K;iFAGpB,yB;MAAA,iE;MAAA,uC;QAOb,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAGB,yB;UAAM,IAAI,UAAU,OA AV,CAAJ,C;YAAwB,OAAO,O;QACrD,MAAM,gCAAuB,sDAAvB,C;O;KARV,C;kGAWA,yB;MAAA,iE;MAAA,uC;QAWW,Q;QAAA,+B;UAcS,U;UAAA,6B;UAAhB,OAAGB,gBAAhB,C;YAAgB,2B;YACZ,aAfwB,SAeX,CAAU,OA AV,C;YACb,IAAI,cAAJ,C;cACI,8BAAO,M;cAAP,gC;UAGR,8BAAO,I;QApBA,kC;QAAA,iB;UAAmC,MAAM,gCAAuB,iEAAvB,C;SAAhD,OAAO,I;O;KAXX,C;8GAcA,gC;MAWoB,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,aAAa,UAAU,OA AV,C;QACb,IAAI,cAAJ,C;UACI,OAAO,M;MAGf,OAAO,I;K;IAGX,mC;MAMI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAA d,C;QACI,OAAO,I;MACX,OAAO,QAAS,O;K;6FAGpB,gC;MAMoB,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,IAAI,UAAU,OA AV,CAAJ,C;UAAwB,OAAO,O;MACrD,OAAO,I;K;IAGX,wC;MAOiB,Q;MADb,YAAY,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,mBAAmB,KAA nB,C;QACA,IAAI,gBAAW,IAAX,CAAJ,C;UACI,OAAO,K;QACX,qB;MAEJ,OAAO,E;K;+FAGX,yB;MAAA,wE;MAAA,uC;QAOiB,Q;QADb,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,mBAAmB,KAA nB,C;UACA,IAAI,UAAU,IAAV,CAAJ,C;YACI,OAAO,K;UACX,qB;QAEJ,OAAO,E;O;KAbX,C;6FAGBA,yB;MAAA,wE;MAAA,uC;QAQiB,Q;QAFb,gBAAGB,E;QACHB,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,mBAAmB,KAA nB,C;UACA,IAAI,UAAU,IAAV,CAAJ,C;YACI,YAAY,K;UACHB,qB;QAEJ,OAAO,S;O;KAdX,C;IAiBA,4B;MAUI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAA d,C;QACI,MAAM,2BAAuB,oBAAvB,C;MACV,WAAW,QAAS,O;MACpB,OAAO,QAAS,UAAhB,C;QACI,OAAO,QAAS,O;MACpB,OAAO,I;K;+EAGX,yB;MAAA,iE;MAAA,gB;MAAA,8B;MAAA,uC;QAYoB,UAQT,M;QAVP,WAAe,I;QACf,YAAY,K;QACI,2B;QAAhB,OAAGB,cAAhB,C;UAGB,yB;UACZ,IAAI,UAAU,OA AV,CAAJ,C;YACI,OAAO,O;YACP,QAAQ,I;QAGhB,IAAI,CAAC,KAAL,C;UAA Y,MAAM,gCAAuB,sDAAvB,C;QAEIB,OAAO,2E;O;KApBX,C;IAuBA,4C;MAQiB,Q;MAFb,gBAAGB,E;MACHB,YAAY,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,mBAAmB,KAA nB,C;QACA,IAAI,gBAAW,IAAX,CAAJ,C;UACI,YAAY,K;QACHB,qB;MAEJ,OAAO,S;K;IAGX,kC;MAQI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAA d,C;QACI,OAAO,I;MACX,WAAW,QAAS,O;MACpB,OAAO,QAAS,UAAhB,C;QACI,OAAO,QAAS,O;MACpB,OAAO,I;K;2FAGX,gC;MASoB,Q;MADhB,WAAe,I;MACC,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,IAAI,UAAU,OA AV,CAAJ,C;UACI,OAAO,O;MAGf,OAAO,I;K;IAGX,8B;MAMI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAA d,C;QACI,MAAM,2BAAuB,oBAAvB,C;MACV,aAAa,QAAS,O;MACTb,IAAI,QAAS,UAA b,C;QACI,MAAM,gCAAYB,qCAAZB,C;MACV,OAAO,M;K;mFAGX,yB;MAAA,kF;MAAA,iE;MAAA,gB;MAAA,8B;MAAA,uC;QAQoB,UAST,M;QAXP,aAAiB,I;QACjB,YAAY,K;QACI,2B;QAAhB,OAAGB,cAAhB,C;U

AAgB,yB;UACZ,IAAI,UAAU,OAAV,CAAJ,C;YACI,IAAI,KAAJ,C;cAAW,MAAM,8BAAYB,mDAAzB,C;YACj  
B,SAAS,O;YACT,QAAQ,I;;QAGhB,IAAI,CAAC,KAAL,C;UAAy,MAAM,gCAAuB,sDAAvB,C;QAEIB,OAAO,  
6E;O;KAjBX,C;IAoBA,oC;MAMI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QACI,OAAO,I;MACX,aAAa,Q  
AAS,O;MACtB,IAAI,QAAS,UAAAb,C;QACI,OAAO,I;MACX,OAAO,M;K;+FAGX,gC;MAQoB,Q;MAFhB,aAAi  
B,I;MACjB,YAAy,K;MACI,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,IAAI,UAAU,OAAV,CAAJ,C;U  
ACI,IAAI,KAAJ,C;YAAW,OAAO,I;UACIB,SAAS,O;UACT,QAAQ,I;;MAGhB,IAAI,CAAC,KAAL,C;QAAy,O  
AAO,I;MACnB,OAAO,M;K;IAGX,8B;MAWW,Q;MhBhXP,IAAI,EgB+WI,KAAK,ChB/WT,CAAJ,C;QACI,cgB  
8Wc,sD;QhB7Wd,MAAM,gCAAyB,OAAQ,WAAjC,C;OgB+WN,UAAK,CAAL,C;QAAU,gB;WACV,+C;QAAiC  
,OAAAL,SAAK,cAAK,CAAL,C;;QACzB,wBAaA,SAAb,EAAmB,CAAnB,C;MAHZ,W;K;IAOJ,2C;MAQI,OAAO,  
sBAAkB,SAAIB,EAAwB,SAAXB,C;K;IAGX,wC;MAQI,OAAO,sBAAkB,SAAIB,EAAwB,IAAxB,EAA8B,SAA9  
B,C;K;IACqE,iD;MAAA,qB;QAAE,yBAAU,EAAG,MAAb,EAAoB,EAAG,MAAvB,C;O;K;IAAkC,oC;MAAE,O  
AAA,EAAG,M;K;IAXzH,+C;MAWI,OAAO,yBAAqB,sBAAkB,qBAAiB,SAAjB,CAAIB,EAA0C,IAA1C,EAAGD  
,+BAAhD,CAArB,EAAYG,sBAAzG,C;K;oGAGX,yB;MA80BA,wE;MA90BA,oD;QAU1BiB,gB;QADb,YAAy,C;  
QACC,2B;QAAAb,OAAa,cAAAb,C;UAAa,sB;UA50BT,IAAI,UA40BkB,oBAAmB,cAAnB,EAAmB,sBAAnB,UA50  
BIB,EA40B+C,IA50B/C,CAAJ,C;YAA2C,sBA40BQ,IA50BR,C;;QAE/C,OAAO,W;O;KAbX,C;sGAgBA,yB;MA  
AA,8C;MAAA,0C;MAAA,8B;MASkB,qD;QAAA,qB;UAAE,c;S;O;MATpB,sC;QASW,Q;QAAP,OAAO,uCAAO,  
iCAAP,gC;O;KATX,C;0GAYA,4C;MAQoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,  
YAAJ,C;UAAkB,WAAy,WAAI,OAAJ,C;;MACpD,OAAO,W;K;IAGX,2C;MAQI,OAAO,sBAAkB,SAAIB,EAA  
wB,KAAxB,EAA+B,SAA/B,C;K;IAYU,kC;MAAE,iB;K;IATvB,oC;MASW,Q;MAAP,OAAO,4CAAU,oBAAV,k  
C;K;IAGX,mD;MAQoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,eAAJ,C;UAAqB,W  
AAy,WAAI,OAAJ,C;;MACvD,OAAO,W;K;4FAGX,6C;MAQoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAA  
gB,yB;QAAM,IAAI,CAAC,UAAU,OAAV,CAAL,C;UAAyB,WAAy,WAAI,OAAJ,C;;MAC3D,OAAO,W;K;sFA  
GX,6C;MAQoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UA  
AwB,WAAy,WAAI,OAAJ,C;;MAC1D,OAAO,W;K;IAGX,8B;MAWW,Q;MhBzGbp,IAAI,EgBwgBI,KAAK,Ch  
BxgBT,CAAJ,C;QACI,cgBugBc,sD;QhBtgBd,MAAM,gCAAyB,OAAQ,WAAjC,C;OgBwgBN,UAAK,CAAL,C;Q  
AAU,sB;WACV,+C;QAAiC,OAAAL,SAAK,cAAK,CAAL,C;;QACzB,wBAaA,SAAb,EAAmB,CAAnB,C;MAHZ,  
W;K;IAOJ,2C;MAQI,OAAO,sBAAkB,SAAIB,EAAwB,SAAXB,C;K;IAWA,2C;MAAA,8B;K;8CACH,Y;MACI,i  
BAA6B,iBAAZ,gBAAY,C;MACIB,QAAX,UAAW,C;MACX,OAAO,UAAW,W;K;;IAZ9B,6B;MAQI,0C;K;sFAS  
J,yB;MAAA,sD;Mdjfa,sC;MAAA,oC;MAAA,uBAOe,yB;QArEf,8D;eAqEe,4B;UAAA,uB;YAAU,eAAsB,gB;YA  
AtB,OA5Dd,cAAc,SA4DgB,CA5DhB,CAAd,EAA2B,SA4DM,CA5DN,CAA3B,C;W;S;OA4DI,C;Mc0ef,sC;QAU  
I,OAAO,sBdpfP,eAAW,iBcofiB,QdpfjB,CAAX,CcofO,C;O;KAVX,C;0GAaA,yB;MAAA,sD;Md3eA,sC;MAAA,o  
C;MAAA,iCAOe,yB;QAxFf,8D;eAwFe,4B;UAAA,uB;YAAU,eAAsB,gB;YAAtB,OA/Ed,cAAc,SA+EgB,CA/EhB  
,CAAd,EAA2B,SA+EM,CA/EN,CAA3B,C;W;S;OA+EI,C;Mcoef,sC;QAQI,OAAO,sBd5eP,eAAW,2Bc4e2B,Qd5e  
3B,CAAX,Cc4eO,C;O;KARX,C;IAWA,uC;MAQI,OAAO,wBAAW,cAAAX,C;K;IAWA,uE;MAAA,sC;MAAA,4C;  
K;kDACH,Y;MACI,iBAAiC,iBAAhB,oBAAGB,C;MACtB,WAAx,UAAW,EAAS,uBAAT,C;MACX,OAAO,UA  
AW,W;K;;IAZ9B,6C;MAQI,0D;K;wFASJ,yB;MAAA,wE;MAAA,uC;QAaW,kBAAY,oB;QAiFH,Q;QAAA,2B;Q  
AAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAlFsC,SAkFvB,CAAU,OAAV,C;UvBnEnB,wBAAI,IAAK,MAA  
T,EAAgB,IAAK,OAArB,C;;QuBfA,OAoFO,W;O;KAjGX,C;6FAGBA,yB;MAAA,wE;MAAA,yC;QAaW,kBAAc,  
oB;QA8BL,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAy,aA/B4B,WA+BxB,CAAY,OAA  
Z,CAAJ,EAA0B,OAA1B,C;;QA/BhB,OAiCO,W;O;KA9CX,C;6FAGBA,yB;MAAA,wE;MAAA,yD;QAYW,kBA  
Ac,oB;QAiCL,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAy,aAIC4B,WakCxB,CAAY,OA  
AZ,CAAJ,EAlCyC,cAkCf,CAAe,OAAf,CAA1B,C;;QAiChB,OAoCO,W;O;KAhDX,C;igAeA,+C;MAYoB,Q;MA  
AA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,WAAy,aAAI,YAAy,OAAZ,CAAJ,EAA0B,OAA1B,C;;M  
AEhB,OAAO,W;K;igAGX,+D;MAYoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,WAAy,aA  
AI,YAAy,OAAZ,CAAJ,EAA0B,eAAe,OAAf,CAA1B,C;;MAEhB,OAAO,W;K;4FAGX,6C;MAWoB,Q;MAAA,2  
B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,WAAe,UAAU,OAAV,C;QvBnEnB,wBAAI,IAAK,MAAT,EAA  
gB,IAAK,OAArB,C;;MuBqEA,OAAO,W;K;gGAGX,yB;MAAA,wE;MAAA,2C;QAcI,aAAa,oB;QAAGB,Q;QAA  
A,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UAFo,MAGBP,aAAI,OAAJ,EAhBe,aAgBF,CAAc,OAAAd,CAAb,C;;

QAhBhB,OAAuB,M;O;Kaf3B,C;oGakBA,iD;MAYoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAGB,yB;Q  
ACZ,WAAY,aAAI,OAAJ,EAAa,cAAc,OAAc,CAAb,C;;MAEhB,OAAO,W;K;IAGX,gD;MAMiB,Q;MAAA,2B;M  
AAb,OAAa,cAAb,C;QAAa,sB;QACT,WAAY,WAAL,IAAJ,C;;MAEhB,OAAO,W;K;IAGX,gC;MAMI,OAAO,0B  
AAa,cAAb,C;K;IAGX,8B;MAMI,OAA4B,qBAAhB,iBAAL,SAAK,CAAgB,C;K;IAGhC,qC;MAMI,OAAO,0BAA  
a,gBAAb,C;K;IAGX,4B;MAQI,OAAwC,oBAAjC,0BAAa,sBAAb,CAAiC,C;K;IAG5C,0C;MAYI,OAAO,uBAAm  
B,SAAnB,EAAyB,SAAzB,6BAAoC,qB;;OAApC,E;K;IAGX,0C;MAQI,OAAO,uBAAmB,SAAnB,EAAyB,SAAz  
B,6BAAoC,qB;;OAApC,E;K;IAGX,iD;MAaI,OAAO,kBA Ae,SAAf,EAAqB,SAArB,6BAAgC,qB;;OAAhC,E;K;I  
AGX,iD;MAaI,OAAO,kBA Ae,SAAf,EAAqB,SAArB,6BAAgC,qB;;OAAhC,E;K;sGAGX,yB;MAAA,wE;MAAA,  
gD;MAAA,oD;QAaoB,UAC4B,M;QAF5C,YAAY,C;QACI,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,W  
AAW,UAAU,oBAAmB,cAAAnB,EAAmB,sBAAnB,UAAV,EAAuC,OAAvC,C;UACC,OAAZ,WAAY,EAAO,IAA  
P,C;;QAEhB,OAAO,W;O;KAjBX,C;uGAoBA,yB;MAAA,wE;MAAA,gD;MAAA,oD;QAaoB,UAC4B,M;QAF5C,  
YAAY,C;QACI,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAY,UAAU,oBAAmB,cAAAnB,EAAmB,sB  
AAAnB,UAAV,EAAuC,OAAvC,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KAjBX,C;yFAoB  
A,yB;MAAA,gD;MAAA,oD;QUoB,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAY,UAA  
U,OAAV,C;UACC,OAAZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KAdX,C;yFAiBA,yB;MAAA,gD;MAA  
A,oD;QAMoB,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAY,UAAU,OAAV,C;UACC,OA  
AZ,WAAY,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KAVX,C;qFAaA,yB;MAAA,wE;MA6BA,+D;MA7BA,yC;QA  
WW,kBAAU,oB;QA6BD,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,UA9BiD,WA8BvC,CAAY  
,OAAZ,C;UvBjoBP,U;UADP,YuBmoBe,WvBnoBH,WuBmoBwB,GvBnoBxB,C;UACL,IAAI,aAAJ,C;YACH,auB  
ioBuC,gB;YAA5B,WvBhoBX,auBgoBgC,GvBhoBhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UuB6nBA,iB;UACA,I  
AAK,WAAL,OAAJ,C;;QAhCT,OAKCO,W;O;KA7CX,C;qFAcA,yB;MAAA,wE;MAkCA,+D;MAiCA,yD;QAYW,  
kBAAU,oB;QAKCD,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,UAnCiD,WAmCvC,CAAY,OA  
AZ,C;UvBrpBP,U;UADP,YuBupBe,WvBvpBH,WuBupBwB,GvBvpBxB,C;UACL,IAAI,aAAJ,C;YACH,auBqpBu  
C,gB;YAA5B,WvBppBX,auBopBgC,GvBppBhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UuBipBA,iB;UACA,IAAK  
,WArCyD,cAqCrD,CAAE,OAaf,CAAJ,C;;QArCT,OAuCO,W;O;KAnDX,C;yFAeA,yB;MAAA,+D;MAAA,sD;Q  
AWoB,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,UAAU,YAAY,OAAZ,C;UvBjoBP,U;UADP,  
YuBmoBe,WvBnoBH,WuBmoBwB,GvBnoBxB,C;UACL,IAAI,aAAJ,C;YACH,auBioBuC,gB;YAA5B,WvBhoBX  
,auBgoBgC,GvBhoBhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UuB6nBA,iB;UACA,IAAK,WAAL,OAAJ,C;;QAEt,  
OAAO,W;O;KAhBX,C;yFAmBA,yB;MAAA,+D;MAAA,sE;QAYoB,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;U  
AAgB,yB;UACZ,UAAU,YAAY,OAAZ,C;UvBrpBP,U;UADP,YuBupBe,WvBvpBH,WuBupBwB,GvBvpBxB,C;U  
ACL,IAAI,aAAJ,C;YACH,auBqpBuC,gB;YAA5B,WvBppBX,auBopBgC,GvBppBhC,EAAS,MAAT,C;YACA,e;;  
YAEA,c;;UuBipBA,iB;UACA,IAAK,WAAL,eAAe,OAaf,CAAJ,C;;QAEt,OAAO,W;O;KAjBX,C;0FAoBA,yB;M  
AAA,kC;MAAA,4C;MAAA,wE;QAUW,sC;QAAA,8C;O;MAVX,oDAWQ,Y;QAA6C,OAAA,oBAAGB,W;O;MA  
XrE,iDAYQ,mB;QAAoC,gCAAY,OAAZ,C;O;MAZ5C,gF;MAAA,yC;QAUI,2D;O;KAVJ,C;IAGBA,sC;MASI,OA  
AO,yBAAqB,SAArB,EAA2B,SAA3B,C;K;IAGX,4C;MASI,OAAO,gCAA4B,SAA5B,EAakC,SAaIC,C;K;IAGX,  
mD;MASI,OAAoD,gBAA7C,gCAA4B,SAA5B,EAakC,SAaIC,CAA6C,C;K;4GAGxD,yB;MAuNA,wE;MAvNA,  
oD;QAGoiB,gB;QADb,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UAvNsB,U;UAAA,wBAuNT,oBAA  
mB,cAAAnB,EAAmB,sBAAnB,UAvNS,EAuNoB,IAvNpB,W;YAA6C,6B;;QACHF,OAAO,W;O;KAVX,C;8FAaA,  
yB;MAAA,wE;MAAA,oD;QAUIB,UACoC,M;QAFjD,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UAC  
T,WAAY,WAAL,UAAU,oBAAmB,cAAAnB,EAAmB,sBAAnB,UAAV,EAAuC,IAAvC,CAAJ,C;;QACHB,OAAO,  
W;O;KAZX,C;IAeA,4C;MASI,OAA6C,gBAAtC,yBAAqB,SAArB,EAA2B,SAA3B,CAAsC,C;K;8FAGjD,yB;MA  
AA,oD;QA4KoB,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UArKK,U;UAAA,wBAqKQ,OArKR,W;Y  
AAsC,6B;;QAC3D,OAAO,W;O;KARX,C;iFAWA,6C;MAOiB,Q;MAAA,2B;MAAb,OAAa,cAAb,C;QAAa,sB;Q  
ACT,WAAY,WAAL,UAAU,IAAV,CAAJ,C;;MACHB,OAAO,W;K;IAGX,gC;MAOI,OAAO,qBAAiB,SAAjB,C;K;  
IACgB,6B;MAAE,S;K;IAX7B,+B;MAWI,OAAY,aAAL,SAAK,EAAW,eAAX,C;K;IAGhB,2C;MAYI,OAAO,qBA  
AiB,SAAjB,EAAuB,QAAvB,C;K;IAGX,mC;MASiB,Q;MADb,UAAU,sB;MACG,2B;MAAb,OAAa,cAAb,C;QA  
Aa,sB;QAAM,GAAL,WAAL,IAAJ,C;;MACvB,OAAO,G;K;6EAGX,gC;MAQoB,Q;MAAA,2B;MAAhB,OAAgB,c  
AAhB,C;QAAGB,yB;QAAM,IAAI,CAAC,UAAU,OAAV,CAAL,C;UAAyB,OAAO,K;;MACTD,OAAO,I;K;IAGX,

2B;MAQI,OAAO,oBAAW,U;K;6EAGtB,gC;MAQoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAA  
M,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,I;MACrD,OAAO,K;K;IAGX,6B;MAOoB,Q;MADhB,YAAY,C  
;MACl,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,oBAAmB,qBAAnB,EAAMB,KAAAnB,E;MACtB,OA  
AO,K;K;iFAGX,yB;MAAA,wE;MAAA,uC;QAOb,Q;QADhB,YAAY,C;QACl,2B;QAAhB,OAAgB,cAAhB,C;U  
AAgB,yB;UAAM,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,oBAAmB,qBAAnB,EAAMB,KAAAnB,E;QAC9C,OAA  
O,K;O;KARX,C;8EAWA,yC;MAYoB,Q;MADhB,kBAAkB,O;MACF,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB  
;QAAM,cAAc,UAAU,WAAV,EAAbB,OAAvB,C;MACpC,OAAO,W;K;4FAGX,yB;MAAA,wE;MAAA,gD;QAc  
oB,UAAiD,M;QAFjE,YAAY,C;QACZ,kBAAkB,O;QACF,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UAAM,cA  
Ac,UAAU,oBAAmB,cAAAnB,EAAMB,sBAAnB,UAAV,EAAC,WAAvC,EAAdD,OAApD,C;QACpC,OAAO,W;  
O;KafX,C;qFAkBA,6B;MAMoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,OAAO,OAAP,C;  
;K;kGAG1B,yB;MAAA,wE;MAAA,oC;QASiB,UAAgC,M;QAD7C,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;U  
AAa,sB;UAAM,OAAO,oBAAmB,cAAAnB,EAAMB,sBAAnB,UAAP,EAAC,IAApC,C;O;KATvB,C;IAYA,2B;M  
AII,OAAO,uB;K;IAGX,2B;MAII,OAAO,uB;K;IAGX,2B;MAGI,OAAO,uB;K;iFAGX,+B;MAGW,sB;QAYP,eA  
Ae,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,I;UAAp,uB;SACzB,cAAc,QAAS,O;QACvB,IAAI,C  
AAC,QAAS,UAAAd,C;UAAyB,qBAAO,O;UAAp,uB;SACzB,eAhBmB,QAGBJ,CAAS,OAAT,C;UAEX,QAAQ,Q  
AAS,O;UACjB,QAnBe,QAmBP,CAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,  
WAAW,C;QAED,QAAT,QAAS,W;QACIB,qBAAO,O;MazBP,yB;K;6FAGJ,+B;MASI,eAAe,oB;MACf,IAAI,C  
AAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHc,cAAc,QAAS,O;MACvB,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,  
OAAO,O;MACHc,eAAe,SAAS,OAAT,C;QAEX,QAAQ,QAAS,O;QACjB,QAAQ,SAAS,CAAT,C;QACR,IAAI,2  
BAAW,CAAX,KAAJ,C;UACI,UAAU,C;UACV,WAAW,C;MAED,QAAT,QAAS,W;MACIB,OAAO,O;K;iFAG  
X,yB;MAAA,sE;MZpwCA,iB;MYowCA,sC;QAeI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM  
,6B;QAC/B,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C  
;UACR,WZhxCG,MAAO,KYgxCO,QZhxCP,EYgxCiB,CZhxCjB,C;QYkxCd,OAAO,Q;O;KATBX,C;iFayBA,yB;  
MAAA,sE;MZxyCA,iB;MYwyCA,sC;QAeI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;Q  
AC/B,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UAC  
R,WZpzCG,MAAO,KYozCO,QZpzCP,EYozCiB,CZpzCjB,C;QYszCd,OAAO,Q;O;KATBX,C;iFayBA,yB;MAA  
A,sE;MAAA,sC;QAaI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;QAC/B,eAAe,SAAS,Q  
AAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,IAAI,2BAAW,CA  
AX,KAAJ,C;YACI,WAAW,C;QAGnB,OAAO,Q;O;KATBX,C;6FayBA,yB;MZ/0CA,iB;MY+0CA,sC;QAaI,eAA  
e,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,OAAO,I;QACHc,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,  
QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,WZz1CG,MAAO,KYy1CO,QZz1CP,EYy1CiB,C  
Zz1CjB,C;QY21Cd,OAAO,Q;O;KApBX,C;6FAuBA,yB;MZj3CA,iB;MYi3CA,sC;QAaI,eAAe,oB;QACf,IAAI,C  
AAC,QAAS,UAAAd,C;UAAyB,OAAO,I;QACHc,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;U  
ACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,WZ33CG,MAAO,KY23CO,QZ33CP,EY23CiB,CZ33CjB,C;QY63Cd  
,OAAO,Q;O;KApBX,C;6FAuBA,+B;MAWI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MA  
ChC,eAAe,SAAS,QAAS,OAAIB,C;MACf,OAAO,QAAS,UAAhB,C;QACI,QAAQ,SAAS,QAAS,OAAIB,C;QAC  
R,IAAI,2BAAW,CAAX,KAAJ,C;UACI,WAAW,C;MAGnB,OAAO,Q;K;yFAGX,yB;MAAA,sE;MAAA,kD;QAa  
I,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;QAC/B,eAAe,SAAS,QAAS,OAAIB,C;QACf,  
OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CA  
AIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;QAGnB,OAAO,Q;O;KATBX,C;qGayBA,2C;MAWI,eAAe,oB;  
MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHc,eAAe,SAAS,QAAS,OAAIB,C;MACf,OAAO,QA  
AS,UAAhB,C;QACI,QAAQ,SAAS,QAAS,OAAIB,C;QACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,  
GAakC,CAAtC,C;UACI,WAAW,C;MAGnB,OAAO,Q;K;IAGX,iC;MASI,eAAe,oB;MACf,IAAI,CAAC,QAAS,  
UAAAd,C;QAAyB,OAAO,I;MACHc,UAAU,QAAS,O;MACnB,OAAO,QAAS,UAAhB,C;QACI,QAAQ,QAAS,O;  
QACjB,MZ18CG,MAAO,KY08CE,GZ18CF,EY08CO,CZ18CP,C;MY48Cd,OAAO,G;K;IAGX,iC;MASI,eAAe,o  
B;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHc,UAAU,QAAS,O;MACnB,OAAO,QAAS,UAAh  
B,C;QACI,QAAQ,QAAS,O;QACjB,MZx+CG,MAAO,KYw+CE,GZx+CF,EYw+CO,CZx+CP,C;MY0+Cd,OAA  
O,G;K;IAGX,iC;MAOI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHc,UAAU,QAAS,

O;MACnB,OAAO,QAAS,UAAhB,C;QACI,QAAQ,QAAS,O;QACjB,IAAI,sBAAM,CAAN,KAAJ,C;UAAa,MAA  
M,C;;MAEvB,OAAO,G;K;IAGX,2C;MAGI,OAAO,4BAAC,UAAAd,C;K;IAGX,iD;MAOI,eAAe,oB;MACf,IAAI,C  
AAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHC,UAAU,QAAS,O;MACnB,OAAO,QAAS,UAAhB,C;QACI,QAA  
Q,QAAS,O;QACjB,IAAI,UAAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MA  
E9C,OAAO,G;K;IAGX,2B;MAII,OAAO,uB;K;IAGX,2B;MAII,OAAO,uB;K;IAGX,2B;MAGI,OAAO,uB;K;iFAG  
X,+B;MAGW,sB;;QAYP,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,I;UAAP,uB;SACzB,cA  
Ac,QAAS,O;QACvB,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,qBAAO,O;UAAP,uB;SACzB,eAhBmB,QAaBJ,CAA  
S,OAAT,C;;UAEX,QAAQ,QAAS,O;UACjB,QAnBe,QAmBP,CAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAA  
J,C;YACI,UAAU,C;YACV,WAAW,C;;QAED,QAAT,QAAS,W;QACIB,qBAAO,O;;;MAzBP,yB;K;6FAGJ,+B;M  
ASI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHC,cAAc,QAAS,O;MACvB,IAAI,CAA  
C,QAAS,UAAAd,C;QAAyB,OAAO,O;MACHC,eAAe,SAAS,OAAT,C;;QAEX,QAAQ,QAAS,O;QACjB,QAAQ,SA  
AS,CAAT,C;QACR,IAAI,2BAAW,CAAX,KAAJ,C;UACI,UAAU,C;UACV,WAAW,C;;MAED,QAAT,QAAS,W;  
MACIB,OAAO,O;K;iFAGX,yB;MAAA,sE;MZj3CA,iB;MYi3CA,sC;QAeI,eAAe,oB;QACf,IAAI,CAAC,QAAS,U  
AAd,C;UAAyB,MAAM,6B;QAC/B,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,  
SAAS,QAAS,OAAIB,C;UACR,WZ73CG,MAAO,KY63CO,QZ73CP,EY63CiB,CZ73CjB,C;;QY+3Cd,OAAO,Q;  
O;KAtBX,C;iFAyBA,yB;MAAA,sE;MZr5CA,iB;MYq5CA,sC;QAeI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAA  
d,C;UAAyB,MAAM,6B;QAC/B,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SA  
AS,QAAS,OAAIB,C;UACR,WZj6CG,MAAO,KYi6CO,QZj6CP,EYi6CiB,CZj6CjB,C;;QYm6Cd,OAAO,Q;O;KAtB  
X,C;iFAyBA,yB;MAAA,sE;MAAA,sC;QAaI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;  
QAC/B,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UA  
CR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAtBX,C;6FAyBA,yB;MZ57CA,iB;M  
Y47CA,sC;QAaI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,OAAO,I;QAChC,eAAe,SAAS,QAAS,OA  
AIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,WZt8CG,MAAO,KYs8CO,  
QZt8CP,EYs8CiB,CZt8CjB,C;;QYw8Cd,OAAO,Q;O;KApBX,C;6FAuBA,yB;MZ99CA,iB;MY89CA,sC;QAaI,eA  
Ae,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,OAAO,I;QAChC,eAAe,SAAS,QAAS,OAAIB,C;QACf,OAA  
O,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,WZx+CG,MAAO,KYw+CO,QZx+CP,EYw+Ci  
B,CZx+CjB,C;;QY0+Cd,OAAO,Q;O;KApBX,C;6FAuBA,+B;MAWI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAA  
d,C;QAAyB,OAAO,I;MACHC,eAAe,SAAS,QAAS,OAAIB,C;MACf,OAAO,QAAS,UAAhB,C;QACI,QAAQ,SAAS,  
QAAS,OAAIB,C;QACR,IAAI,2BAAW,CAAX,KAAJ,C;UACI,WAAW,C;;MAGnB,OAAO,Q;K;yFAGX,yB;MA  
AA,sE;MAAA,kD;QAaI,eAAe,oB;QACf,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,6B;QAC/B,eAAe,SAAS,  
QAAS,OAAIB,C;QACf,OAAO,QAAS,UAAhB,C;UACI,QAAQ,SAAS,QAAS,OAAIB,C;UACR,IAAI,UAAW,SA  
AQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAtBX,C;qGAyB  
A,2C;MAWI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHC,eAAe,SAAS,QAAS,OAAI  
B,C;MACf,OAAO,QAAS,UAAhB,C;QACI,QAAQ,SAAS,QAAS,OAAIB,C;QACR,IAAI,UAAW,SAAQ,QAAR,E  
AAkB,CAAIB,CAAX,GAakC,CAAtC,C;UACI,WAAW,C;;MAGnB,OAAO,Q;K;IAGX,iC;MASI,eAAe,oB;MAC  
f,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHC,UAAU,QAAS,O;MACnB,OAAO,QAAS,UAAhB,C;QA  
CI,QAAQ,QAAS,O;QACjB,MZvjDG,MAAO,KYujDE,GZvjDF,EYujDO,CZvjDP,C;;MYyjDd,OAAO,G;K;IAGX,  
iC;MASI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHC,UAAU,QAAS,O;MACnB,OA  
AO,QAAS,UAAhB,C;QACI,QAAQ,QAAS,O;QACjB,MZrIDG,MAAO,KYqlIDE,GZrIDF,EYqlIDO,CZrIDP,C;;MY  
ulDd,OAAO,G;K;IAGX,iC;MAOI,eAAe,oB;MACf,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHC,UAA  
U,QAAS,O;MACnB,OAAO,QAAS,UAAhB,C;QACI,QAAQ,QAAS,O;QACjB,IAAI,sBAAM,CAAN,KAAJ,C;UA  
Aa,MAAM,C;;MAEvB,OAAO,G;K;IAGX,2C;MAGI,OAAO,4BAAC,UAAAd,C;K;IAGX,iD;MAOI,eAAe,oB;MAC  
f,IAAI,CAAC,QAAS,UAAAd,C;QAAyB,OAAO,I;MACHC,UAAU,QAAS,O;MACnB,OAAO,QAAS,UAAhB,C;QA  
CI,QAAQ,QAAS,O;QACjB,IAAI,UAAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM  
,C;;MAE9C,OAAO,G;K;IAGX,4B;MAQI,OAAO,CAAC,oBAAW,U;K;+EAGvB,gC;MAQoB,Q;MAAA,2B;MAA  
hB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,K;;MACrD,OAAO,I;K;  
IAUI,uC;MAAA,qB;QACP,eAAO,EAAP,C;QAAA,OACA,E;O;K;IATR,sC;MAOI,OAAO,kBAAI,qBAAJ,C;K;IA  
eW,8C;MAAA,iC;QACd,eAAO,KAAP,EAAC,OAAd,C;QAAA,OACA,O;O;K;IAXR,6C;MASI,OAAO,wBAAW,4

BAAX,C;K;kFAMX,yB;MAAA,4F;MAAA,uC;QAEI,eAAe,SAAK,W;QACpB,IAAI,CAAC,QAAS,UAAAd,C;UAA  
yB,MAAM,mCAA8B,kCAA9B,C;QAC/B,kBAAqB,QAAS,O;QAC9B,OAAO,QAAS,UAAhB,C;UACI,cAAc,UA  
AU,WAAV,EAAuB,QAAS,OAAhC,C;;QAEIB,OAAO,W;O;KArBX,C;gGawBA,yB;MAAA,4F;MAAA,wE;MA  
AA,uC;QAOBmD,Q;QAL/C,eAAe,SAAK,W;QACpB,IAAI,CAAC,QAAS,UAAAd,C;UAAyB,MAAM,mCAA8B,k  
CAA9B,C;QAC/B,YAAy,C;QACZ,kBAAqB,QAAS,O;QAC9B,OAAO,QAAS,UAAhB,C;UACI,cAAc,UAAU,oB  
AAmB,YAAnB,EAAMb,oBAAnB,QAAs,EAAuC,WAAvC,EAAoD,QAAS,OAA7D,C;;QAEIB,OAAO,W;O;KAt  
BX,C;4GAyBA,yB;MAAA,wE;MAAA,uC;QAOBmD,Q;QAL/C,eAAe,SAAK,W;QACpB,IAAI,CAAC,QAAS,UA  
Ad,C;UAAyB,OAAO,I;QACChC,YAAy,C;QACZ,kBAAqB,QAAS,O;QAC9B,OAAO,QAAS,UAAhB,C;UACI,cA  
Ac,UAAU,oBAAMb,YAAnB,EAAMb,oBAAnB,QAAs,EAAuC,WAAvC,EAAoD,QAAS,OAA7D,C;;QAEIB,OA  
AO,W;O;KATBX,C;8FAyBA,gC;MAGBI,eAAe,SAAK,W;MACpB,IAAI,CAAC,QAAS,UAAAd,C;QAAYB,OAAO,I  
;MACHC,kBAAqB,QAAS,O;MAC9B,OAAO,QAAS,UAAhB,C;QACI,cAAc,UAAU,WAAV,EAAuB,QAAS,OAA  
hC,C;;MAEIB,OAAO,W;K;IAoBS,2I;MAAA,wC;MAAA,6B;MAAA,yB;MAAA,8C;MAAA,gD;MAAA,kD;MAA  
A,wB;MAAA,+B;MAAA,kC;K;;;sDAAA,Y;;;;cACZ,gB;8BAAA,iCAAM,0BAAN,O;kBAAA,2C;uBAAA,yB;cA  
AA,Q;;;;uCACkB,0B;cACF,wD;cAAhB,gB;;;cAAA,KAAGb,yBAhB,C;gBAAA,gB;;;cAAGb,oC;cACZ,yBAAc,  
6BAAU,sBAAV,EAAuB,OAAvB,C;cACd,gB;8BAAA,iCAAM,sBAAN,O;kBAAA,2C;uBAAA,yB;cAAA,Q;;cAF  
J,gB;;;cAIJ,W;;;;K;IAPgB,wF;MAAA,yD;uBAAA,+H;YAAA,S;iBAAA,Q;;iBAAA,uB;O;K;IAjBpB,sD;M  
AiBI,OAAO,SAAS,iDAAT,C;K;IA4BS,yJ;MAAA,wC;MAAA,6B;MAAA,yB;MAAA,8C;MAAA,8D;MAAA,kD;  
MAAA,wB;MAAA,yB;MAAA,+B;MAAA,kC;K;;;6DAAA,Y;;;;kBAKmC,I;cAJ/C,gB;8BAAA,iCAAM,0BAAN,  
O;kBAAA,2C;uBAAA,yB;cAAA,Q;;;;iCACY,C;uCACM,0B;cACF,+D;cAAhB,gB;;;cAAA,KAAGb,yBAhB,C;g  
BAAA,gB;;;cAAGb,oC;cACZ,yBAAc,6BAAU,oBAAMb,uBAAnB,EAAMb,+BAAnB,QAAs,EAAuC,sBAAvC,E  
AAoD,OAApD,C;cACd,gB;8BAAA,iCAAM,sBAAN,O;kBAAA,2C;uBAAA,yB;cAAA,Q;;cAFJ,gB;;cAIJ,W;;;;  
;K;IARgB,sG;MAAA,yD;uBAAA,6I;YAAA,S;iBAAA,Q;;iBAAA,uB;O;K;IALpB,6D;MAkBI,OAAO,SAAS,  
wDAAT,C;K;IA2BS,4H;MAAA,wC;MAAA,6B;MAAA,yB;MAAA,oD;MAAA,kD;MAAA,4B;MAAA,+B;MAA  
A,kC;K;;;wDAAA,Y;;;;oCACG,wC;cACf,IAAI,mBAAS,UAAb,C;yCACyB,mBAAS,O;gBAC9B,gB;gCAAA,iC  
AAM,sBAAN,O;oBAAA,2C;yBAAA,yB;gBAAA,Q;;gBAFJ,gB;;;;cAGI,gB;;;cAAA,KAAs,mBAAS,UAAhB,C  
;gBAAA,gB;;;cACI,yBAAc,6BAAU,sBAAV,EAAuB,mBAAS,OAAhC,C;cACd,gB;8BAAA,iCAAM,sBAAN,O;k  
BAAA,2C;uBAAA,yB;cAAA,Q;;cAFJ,gB;;;cAHJ,gB;;;cAQJ,W;;;;K;IAVgB,yE;MAAA,yD;uBAAA,gH;Y  
AAA,S;iBAAA,Q;;iBAAA,uB;O;K;IAhBpB,+C;MAGBI,OAAO,SAAS,0CAAT,C;K;IA6BS,0I;MAAA,wC;MAAA  
,6B;MAAA,yB;MAAA,kE;MAAA,kD;MAAA,4B;MAAA,+B;MAAA,yB;MAAA,kC;K;;;+DAAA,Y;;;;cAOuC,Q  
;oCANpC,+C;cACf,IAAI,mBAAS,UAAb,C;yCACyB,mBAAS,O;gBAC9B,gB;gCAAA,iCAAM,sBAAN,O;oBAA  
A,2C;yBAAA,yB;gBAAA,Q;;gBAFJ,gB;;;;iCAGgB,C;cACZ,gB;;;cAAA,KAAs,mBAAS,UAAhB,C;gBAAA,g  
B;;cACI,yBAAc,6BAAU,oBAAMb,uBAAnB,EAAMb,+BAAnB,QAAs,EAAuC,sBAAvC,EAAoD,mBAAS,OA  
A7D,C;cACd,gB;8BAAA,iCAAM,sBAAN,O;kBAAA,2C;uBAAA,yB;cAAA,Q;;cAFJ,gB;;;cAJJ,gB;;;cASJ,W;;;;  
;K;IAXgB,uF;MAAA,yD;uBAAA,8H;YAAA,S;iBAAA,Q;;iBAAA,uB;O;K;IAhBpB,sD;MAGBI,OAAO,SA  
S,iDAAT,C;K;IAcX,+C;MAkBI,OAAO,yBAAy,OAAZ,EAAqB,SAArB,C;K;IAGX,sD;MAmBI,OAAO,gCAAmB  
,OAAAnB,EAA4B,SAA5B,C;K;gFAGX,+B;MASoB,Q;MADhB,UAAe,C;MACC,2B;MAAhB,OAAGb,cAAhB,C;Q  
AAGb,yB;QACZ,YAAO,SAAS,OAAT,CAAP,I;;MAEJ,OAAO,G;K;4FAGX,+B;MASoB,Q;MADhB,UAAkB,G;  
MACF,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QACZ,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;iFAGX,+  
B;MAYoB,Q;MADhB,UAAoB,C;MACJ,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QACZ,OAAO,SAAS,OAAT  
,C;;MAEX,OAAO,G;K;iFAGX,+B;MAYoB,Q;MADhB,UAAe,C;MACC,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,  
yB;QACZ,YAAO,SAAS,OAAT,CAAP,I;;MAEJ,OAAO,G;K;iFAGX,yB;MAAA,SAWoB,gB;MAXpB,sC;QAYoB  
,Q;QADhB,Y;QACgB,2B;QAAhB,OAAGb,cAAhB,C;UAGb,yB;UACZ,cAAO,SAAS,OAAT,CAAP,C;;QAEJ,O  
AAO,G;O;KAFx,C;iFAkBA,yB;M3B15DA,6B;M2B05DA,sC;QAaoB,Q;QADhB,U3B55DmC,c2B45DnB,C3B55  
DmB,C;Q2B65DnB,2B;QAAhB,OAAGb,cAAhB,C;UAGb,yB;UACZ,M3BhuEiD,c2BguEjD,G3BhuE2D,KAAX,  
G2BguEzD,SAAS,OAAT,C3BhuEoE,KAAX,IAAf,C;;Q2BkuErD,OAAO,G;O;KAhBX,C;iFAMBA,yB;MX16DA,  
+B;MW06DA,sC;QAaoB,Q;QADhB,UX36DqC,eAAW,oBW26D/B,CX36D+B,CAAX,C;QW46DrB,2B;QAAhB,  
OAAGb,cAAhB,C;UAGb,yB;UACZ,MXhVEmD,eWgvEnD,GXhVE8D,KAAX,KWgvE5D,SAAS,OAAT,CXhVE  
uE,KAAX,CAAhB,C;;QWkvEvD,OAAO,G;O;KAhBX,C;IAyBe,oD;MAAA,qB;QAAE,e;UAAAM,MAAM,gCAAy

B,2BAAwB,mBAAxB,MAAzB,C;SAAZ,S;O;K;IANjB,qC;MAMI,OAAO,kBAAl,gCAAJ,C;K;IAGX,oC;MAaI,OAAO,sBAAS,IAAT,EA Ae,IAAf,EAAsC,IAAtC,C;K;IAGX,+C;MakBI,OAAO,sBAAS,IAAT,EA Ae,IAAf,EAAsC,IAAtC,EA AwD,SAAxD,C;K;IASA,0D;MAAA,4B;MAAA,sC;K;IAG0B,+E;MAAA,qB;QAAE,IAAI,CAAC,iBAAD,IAAY,WAAM,eAAN,CAAhB,C;UAAiC,oBAAU,I;UAA3C,OAAiD,K;UAAjD,OAA8D,I;O;K;6CAF7F,Y;MACI,kBAAc,KAAd,C;MACA,OAAkB,SAAX,eAAW,EAAO,kEAAP,CAA8E,W;K;IAT5G,qC;MAMI,kD;K;IASBO,6D;MAAA,wC;MAAA,4B;K;IAG6B,8D;MAAA,qB;QAAE,OAAM,aAAN,mB;O;K;+CAFIC,Y;MACI,YAAqB,8BAAT,qBAAS,C;MACrB,OAAkB,YAAX,eAAW,EAAU,4CAAV,CAA0B,W;K;IAjBxD,sC;MAaI,IAAI,Q9B80KG,YAAQ,C8B90Kf,C;QAAwB,OAAO,S;MAC/B,qD;K;IAqBO,6D;MAAA,wC;MAAA,4B;K;IAMiC,8D;MAAA,qB;QAAE,OAAM,aAAN,mB;O;K;+CALtC,Y;MACI,YAAqB,4BAAT,qBAAS,C;MACrB,IAAI,KAAM,UAAV,C;QACI,OAAO,eAAW,W;;QAEIB,OAAkB,YAAX,eAAW,EAAU,4CAAV,CAA0B,W;K;IANB5D,sC;MAaI,qD;K;IAwBO,6D;MAAA,wC;MAAA,4B;K;IAMiC,8D;MAAA,qB;QAAE,OAAM,aAAN,mB;O;K;+CALtC,Y;MACI,YAAqB,8BAAT,qBAAS,C;MACrB,IAAI,KAAM,UAAV,C;QACI,OAAO,eAAW,W;;QAEIB,OAAkB,YAAX,eAAW,EAAU,4CAAV,CAA0B,W;K;IANB5D,sC;MAaI,qD;K;8FAWJ,yB;MAAA,4C;MAAA,qC;QAOI,OAAO,iBAA M,OAAAN,C;O;KAPX,C;wFAUA,yB;MAAA,+D;MAAA,6B;MAAA,uC;QAYoB,Q;QAFhB,YAAY,gB;QACZ,aA Aa,gB;QACG,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,IAAI,UAAU,OAAV,CAAJ,C;YACI,KAAM,WAAI,OAAJ,C;;YAEN,MAAO,WAAl,OAAJ,C;;QAGf,OAAO,cAAK,KAAL,EAAY,MAAZ,C;O;KANBX,C;IASB A,oC;MAMI,OAA6C,UAAtC,YAAW,SAAX,EAAiB,YAAW,OAA X,EAAjB,EAAsC,C;K;IAGjD,qC;MASI,OAA Y,OAAAL,SAAK,EAAC,OAAT,QAAS,CAAd,C;K;IAGhB,qC;MASI,OAA+C,UAAxC,YAAW,SAAX,EAA0B,aA AT,QAAS,CAA1B,EA AwC,C;K;IAGnD,sC;MASI,OAAkC,UAA3B,YAAW,SAAX,EAAiB,QAAjB,EAA2B,C;K;4FAGtC,yB;MAAA,0C;MAAA,qC;QAOI,OAAO,gBAAK,OAAL,C;O;KAPX,C;IAUA,2D;MAGb+C,oB;QAAA, OAA Y,C;MAAG,8B;QAAA,iBAA0B,K;MACpF,OAAO,8BAAiB,IAAjB,EAAuB,IAAvB,EAA6B,cAA7B,EAA2 D,KAA3D,C;K;IAGX,sE;MakBkD,oB;QAAA,OAA Y,C;MAAG,8B;QAAA,iBAA0B,K;MACvF,OAAwE,OAAjE ,8BAAiB,IAAjB,EAAuB,IAAvB,EAA6B,cAA7B,EAA2D,IAA3D,CAAiE,EAAl,SAAJ,C;K;IAYpC,4B;MAAY,c AAM,EAAN,C;K;IATpD,kC;MASI,OAAO,oBAAgB,SAAhB,EAAsB,KAAtB,EAA6B,UAA7B,C;K;IAGX,6C;M AUI,OAAO,oBAAgB,SAAhB,EAAsB,KAAtB,EAA6B,SAA7B,C;K;IAcY,kC;MAAU,aAAK,CAAL,C;K;IAXjC,k C;MAWI,OAAO,yBAAY,kBAAZ,C;K;IAeiB,wH;MAAA,wC;MAAA,6B;MAAA,yB;MAAA,gD;MAAA,kD;MA AA,4B;MAAA,2B;MAAA,wB;MAAA,kC;K;;sDAAA,Y;;;oCACL,sC;CACf,IAAI,CAAC,mBAAS,UAA d,C;gB AAyB,M;;gBAAzB,gB;;;;mCACc,mBAAS,O;cACvB,gB;;cAAA,KAAO,mBAAS,UAAhB,C;gBAAA,gB;;gCA Ce,mBAAS,O;cACpB,gB;8BAAA,iCAAM,6BAAU,kBAAV,EAAMb,eAAnB,CAAN,O;kBAAA,2C;uBAAA,yB;c AAA,Q;;cACA,qBAAU,e;cAHd,gB;;cAKJ,W;;;K;IATwB,uE;MAAA,yD;uBAAA,4G;YAAA,S;iBAAA,Q; ;iBAAA,uB;O;K;IAZ5B,6C;MAYI,OAAO,SAAS,0CAAT,C;K;IAYX,8F;MAU6D,yB;QAAA,YAA0B,I;MAAM,s B;QAAA,SAAuB,E;MAAI,uB;QAAA,UAAwB,E;MAAI,qB;QAAA,QAAa,E;MAAI,yB;QAAA,YAA0B,K;MAA O,yB;QAAA,YAAoC,I;MAGtN,Q;MAFhB,MAAO,gBAAO,MAAP,C;MACP,YAA Y,C;MACI,2B;MAAhB,OAAg B,cAAhB,C;QAAgB,yB;QACZ,IAAI,iCAAU,CAAd,C;UAAiB,MAAO,gBAAO,SAAP,C;QACxB,IAAI,QAAQ,C AAR,IAAa,SAAS,KAA1B,C;UACW,gBAAP,MAAO,EAAC,OAAd,EAAuB,SAAvB,C;;UACJ,K;;MAEX,IAAI,S AAS,CAAT,IAAc,QAAQ,KAA1B,C;QAAiC,MAAO,gBAAO,SAAP,C;MACxC,MAAO,gBAAO,OAAP,C;MACP ,OAAO,M;K;IAGX,4F;MAUwC,yB;QAAA,YAA0B,I;MAAM,sB;QAAA,SAAuB,E;MAAI,uB;QAAA,UAAwB,E ;MAAI,qB;QAAA,QAAa,E;MAAI,yB;QAAA,YAA0B,K;MAAO,yB;QAAA,YAAoC,I;MACjN,OAAO,oBAAO,s BAAP,EA AwB,SAAxB,EAAMc,MAAnC,EAA2C,OAA3C,EAAoD,KAAPd,EAA2D,SAA3D,EAAsE,SAAtE,CA AiF,W;K;IAOxE,8C;MAAA,mB;QAAE,OAAA,eAAK,W;O;K;IAJ3B,kC;MAII,oCAAgB,8BAAhB,C;K;2FAGJ,q B;MAKI,OAAO,S;K;IAGX,+B;MASoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MACD,2B;MAAhB,OAAgB,c AAhB,C;QAAgB,yB;QACZ,OAAO,O;QACP,oBAAmB,qBAAnB,EAAMb,KAAAnB,E;;MAEJ,OAAW,UAA S,CA Ab,GAAgB,wCAAO,IAAvB,GAAgC,MAAM,K;K;IAGjD,+B;MASoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C; MACD,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,OAAO,O;QACP,oBAAmB,qBAAnB,EAAMb,KAAAnB ,E;;MAEJ,OAAW,UAA S,CAAb,GAAgB,wCAAO,IAAvB,GAAgC,MAAM,K;K;IAGjD,+B;MASoB,Q;MAFhB,U AAKB,G;MACIB,YAAiB,C;MACD,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,OAAO,O;QACP,oBAAm B,qBAAnB,EAAMb,KAAAnB,E;;MAEJ,OAAW,UAA S,CAAb,GAAgB,wCAAO,IAAvB,GAAgC,MAAM,K;K;IA GjD,+B;MASoB,Q;MAFhB,UAAkB,G;MACIB,YAAiB,C;MACD,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QA

CZ,OAAO,O;QACP,oBAAmB,qBAAnB,EAAMB,KAAAnB,E;;MAEJ,OAAW,UAAS,CAAb,GAAgB,wCAAO,IAAvB,GAAgC,MAAM,K;K;IAGjD,+B;MASoB,Q;MAFhB,UAakB,G;MACIB,YAAiB,C;MACD,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,OAAO,O;QACP,oBAAmB,qBAAnB,EAAMB,KAAAnB,E;;MAEJ,OAAW,UAAS,CAAb,GAAgB,wCAAO,IAAvB,GAAgC,MAAM,K;K;IAGjD,+B;MASoB,Q;MAFhB,UAakB,G;MACIB,YAAiB,C;MACD,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,OAAO,O;QACP,oBAAmB,qBAAnB,EAAMB,KAAAnB,E;;MAEJ,OAAW,UAAS,CAAb,GAAgB,wCAAO,IAAvB,GAAgC,MAAM,K;K;IAGjD,2B;MAQoB,Q;MADhB,UA Ae,C;MACC,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,YAAO,O;;MAEX,OAAO,G;K;IAGX,2B;MAQoB,Q;MADhB,UA Ae,C;MACC,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,YAAO,O;;MAEX,OAAO,G;K;IAGX,2B;MAQoB,Q;MADhB,UA Ae,C;MACC,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,YAAO,O AAP,I;;MAEJ,OAAO,G;K;IAGX,2B;MAQoB,Q;MADhB,Y;MACgB,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,cAAO,OAAP,C;;MAEJ,OAAO,G;K;IAGX,2B;MAQoB,Q;MADhB,UA AiB,G;MACD,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,OAAO,O;;MAEX,OAAO,G;K;IAGX,2B;MAQoB,Q;MADhB,UAakB,G;MACF,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QACZ,OAAO,O;;MAEX,OAAO,G;K;IC71FX,qC;MAMI,aAAa,qBAAiB,YAAy,cAAZ,CAAjB,C;MACb,kBAAC,KAAD,C;MX4zBgB,Q;MAAA,OW3zBT,SX2zBS,W;MAAhB,OAAGB,cAAhB,C;QAAGB,2B;QAAU,oB;QW3zBK,IAAI,CAAC,SAAD,IAAY,OX2zBX,SW3zBW,UA AhB,C;UAAiC,YAAU,I;UAA3C,mBAAiD,K;;UAAjD,mBAA8D,I;;QX2zBvE,qB;UW3zBD,MX2zBqC,WAAI,SAAJ,C;;MW3zB1D,OAAqB,M;K;IAGzB,sC;MAUI,aAAa,qBAAiB,SAAJB,C;MACN,YAAP,MAAO,EAAU,QAAV,C;MACP,OAAO,M;K;IAGX,sC;MAUI,YAAqB,gCAAT,QAAS,EAAgC,SAAhC,C;MACrB,IAAI,KAAM,UAAV,C;QACI,OAA Y,QAAL,SAAK,C;MACHB,IAAI,yBAAJ,C;QACgB,kBAAY,sB;QXixBZ,Q;QAAA,OWjxBL,SXixBK,W;QA AhB,OAAGB,cAAhB,C;UAAgB,yB;UAAM,IAAI,CWjxBwB,qBXixBb,OWjxBa,CXixB5B,C;YAAyB,WAAy,WAAI,OA AJ,C;;QWjxBvD,OXkxBG,W;OWjxBP,aAAa,qBAAiB,SAAJB,C;MACb,MAAO,mBAAU,KAAV,C;MACP,OAAO,M;K;IAGX,uC;MAUI,aAAa,qBAAiB,SAAJB,C;MACN,YAAP,MAAO,EAAU,QAAV,C;MACP,OAAO,M;K;gGAGX,yB;MAAA,8C;MAAA,qC;QAOI,OAAO,iBAAM,OAAN,C;O;KAPX,C;IAUA,qC;MAMI,aAAa,qBAAiB,YAAy,iBAAO,CAAP,IAAZ,CAAjB,C;MACb,MAAO,gBAAO,SAAP,C;MACP,MAAO,WAAI,OA AJ,C;MACP,OAAO,M;K;IAGX,sC;MAOI,aAAa,qBAAiB,YAAy,SAAK,KAAL,GAAY,QAAS,OAArB,IAAZ,CAAjB,C;MA Cb,MAAO,gBAAO,SAAP,C;MACA,SAAP,MAAO,EAAO,QAAP,C;MACP,OAAO,M;K;IAGX,sC;MAMuD,UA AT,M;MAA1C,aAAa,qBAAiB,YAAy,WAAS,4BAAT,QAAS,CAAT,YAA4C,cAAL,WAAvC,4BAA2D,SAAK,K AAL,GAAY,CAAZ,IAAvE,CAAjB,C;MACb,MAAO,gBAAO,SAAP,C;MACA,OAAP,MAAO,EAAO,QAAP,C; MACP,OAAO,M;K;IAGX,sC;MAOI,aAAa,qBAAiB,YAAy,SAAK,KAAL,GAAY,CAAZ,IAAZ,CAAjB,C;MACb ,MAAO,gBAAO,SAAP,C;MACA,SAAP,MAAO,EAAO,QAAP,C;MACP,OAAO,M;K;8FAGX,yB;MAAA,4C;M AAA,qC;QAOI,OAAO,gBAAK,OAAL,C;O;KAPX,C;InBnIA,oD;MAMuF,wC;K;IANvF,8CAOI,Y;MAAuC,8B;K ;IAP3C,gF;ICGA,oD;MAQuF,wC;K;IARvF,8CASI,Y;MAAuC,8B;K;IAT3C,gF;gGmBYA,yB;MAAA,uD;MAAA, gC;MAAA,iD;QAOI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAsC,qBAAI,KAAJ,CAAtC,GAAsD,uBAA a,KAAb,E;O;KAPjE,C;gGAUA,yB;MAAA,+C;MAAA,mC;QAOI,OAAy,UAAL,SAAK,EAAU,KAAV,C;O;KAP hB,C;0EAUA,yB;MA4EA,6C;MAAA,oC;MAAA,gC;MA5EA,uC;QAOW,sB;;UAyES,Q;UAAA,0B;UAAhB,OA AgB,cAAhB,C;YAAgB,oC;YAAM,IAzEH,SAyEO,CAAU,oBAAV,CAAJ,C;cAAwB,qBAAO,O;cAAP,uB;;UAC9C ,qBAAO,I;;QA1EP,yB;O;KAPJ,C;kFAUA,yB;MAwJA,mD;MAAA,+C;MAAA,oC;MAxJA,uC;QAOW,qB;;UAuJ O,Q;UAAA,OAAa,SAAR,sBAAQ,CAAb,W;UAAAd,OAAC,cAAAd,C;YAAc,uB;YACV,cAAc,qBAAK,KAAL,C;YA Cd,IAzJc,SAyJV,CAAU,oBAAV,CAAJ,C;cAAwB,oBAAO,O;cAAP,sB;;UAE5B,oBAAO,I;;QA3JP,wB;O;KAPJ, C;IAUA,6B;MAKI,ICkOgD,qBAAU,CDIO1D,C;QACI,MAAM,2BAAuB,yBAAvB,C;MACV,OAAO,qBAAK,CA AL,C;K;4EAGX,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,iE;MAAA,uC;QAKoB,Q;QAAA,0B;QA AhB,OA AgB,cAAhB,C;UAAgB,oC;UAAM,IAAI,UAAU,oBAAV,CAAJ,C;YAAwB,OAAO,O;;QACrD,MAAM,gCAAuB, 6DAAvB,C;O;KANV,C;6FASA,yB;MAAA,iE;MAY A,6C;MAAA,oC;MAAA,gC;MAZA,uC;QASW,Q;QAAA,+B ;;UAYS,U;UAAA,4B;UAAhB,OAAGB,gBAAhB,C;YAAgB,sC;YACZ,aAbwB,SAAX,CAAU,oBAAV,C;YACb,IA AI,cAAJ,C;cACI,8BAAO,M;cAAP,gC;;UAGR,8BAAO,I;;QA1BA,kC;QAAA,iB;UAAmC,MAAM,gCAAuB,sEA AvB,C;SAAhD,OAAO,I;O;KATX,C;yGAYA,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,uC;QASoB,Q;QAAA ,0B;QA AhB,OAAGB,cAAhB,C;UAAgB,oC;UACZ,aAAa,UAAU,oBAAV,C;UACb,IAAI,cAAJ,C;YACI,OAAO,M ;;QAGf,OAAO,I;O;KAFx,C;IAkBA,mC;MAII,OckLgD,qBAAU,CDILnD,GA Ae,IAAf,GAAYB,qBAAK,CAAL,C

;K;wFAGpC,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,uC;QAIoB,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;U  
AAgB,oC;UAAM,IAAI,UAAU,oBAAV,CAAJ,C;YAAwB,OAAO,O;;QACrD,OAAO,I;O;KALX,C;mFAQA,yB;  
MAAA,uD;MAAA,gC;MAAA,iD;QAKI,OAAW,SAAS,CAAT,IAAc,SAAS,wBAA3B,GAAc,qBAAI,KAJ,CA  
AtC,GAAcD,uBAAa,KAAb,E;O;KALjE,C;IAQA,uC;MAMI,OAAW,SAAS,CAAT,IAAc,SAAS,2BAA3B,GAAc  
,qBAAI,KAJ,CAAtC,GAAcD,I;K;0FAGjE,yB;MAAA,mD;MAAA,oC;MAAA,uC;QAIkB,gC;QAAA,6B;QAAA,  
mB;QAAA,kB;QAAA,kB;QAAd,0D;UACI,IAAI,UAAU,iCAAK,KAAL,EAAV,CAAJ,C;YACI,OAAO,K;;QAGf,  
OAAO,E;O;KATX,C;wFAYA,yB;MAAA,mD;MAAA,+C;MAAA,oC;MAAA,uC;QAIkB,Q;QAAA,OAAQ,SAAR  
,sBAAQ,CAAR,W;QAAd,OAAc,cAAc,C;UAAc,uB;UACV,IAAI,UAAU,iCAAK,KAAL,EAAV,CAAJ,C;YACI,O  
AAO,K;;QAGf,OAAO,E;O;KATX,C;IAYA,4B;MAQI,ICsHgD,qBAAU,CDtH1D,C;QACI,MAAM,2BAAuB,yBA  
AvB,C;MACV,OAAO,qBAAK,2BAAL,C;K;0EAGX,yB;MAAA,mD;MAAA,+C;MAAA,oC;MAAA,iE;MAAA,u  
C;QAQkB,Q;QAAA,OAAa,SAAR,YAAL,SAAK,CAAQ,CAAb,W;QAAd,OAAc,cAAc,C;UAAc,uB;UACV,cAAc  
,qBAAK,KAAL,C;UACd,IAAI,UAAU,oBAAV,CAAJ,C;YAAwB,OAAO,O;;QAEnC,MAAM,gCAAuB,6DAAvB,  
C;O;KAZV,C;IAeA,kC;MAMI,OC4FgD,qBAAU,CD5FnD,GAAe,IAAf,GAAyB,qBAAK,mBAAS,CAAT,IAAL,C  
;K;sFAGpC,yB;MAAA,mD;MAAA,+C;MAAA,oC;MAAA,uC;QAMkB,Q;QAAA,OAAa,SAAR,YAAL,SAAK,C  
AAQ,CAAb,W;QAAd,OAAc,cAAc,C;UAAc,uB;UACV,cAAc,qBAAK,KAAL,C;UACd,IAAI,UAAU,oBAAV,CA  
AJ,C;YAAwB,OAAO,O;;QAEnC,OAAO,I;O;KAVX,C;8EAaA,yB;MAAA,mC;MAAA,yC;MAAA,4B;QAQI,OA  
AO,kBAAO,cAAP,C;O;KARX,C;IAWA,sC;MAOI,IC0DgD,qBAAU,CD1D1D,C;QACI,MAAM,2BAAuB,yBAAv  
B,C;MACV,OAAO,qBAAI,MAAO,iBAAQ,gBAAR,CAAX,C;K;0FAGX,yB;MAAA,mC;MAAA,qD;MAAA,4B;  
QAOI,OAAO,wBAAa,cAAb,C;O;KAPX,C;IAUA,4C;MAMI,ICqCgD,qBAAU,CDrC1D,C;QACI,OAAO,I;MACX  
,OAAO,qBAAI,MAAO,iBAAQ,gBAAR,CAAX,C;K;IAGX,8B;MAlIB,IAAN,I;MAAA,QAAM,gBAAN,C;aACH,  
C;UAAK,MAAM,2BAAuB,yBAAvB,C;aACX,C;UAAK,4BAAK,CAAL,C;UAAL,K;gBACQ,MAAM,gCAAyB,0  
CAAzB,C;;MAHIB,W;K;8EAOJ,yB;MAAA,6C;MAAA,oC;MAAA,kF;MAAA,gC;MAAA,iE;MAAA,8B;MAAA,  
uC;QAMoB,UAST,M;QAXP,aAAoB,I;QACpB,YAAY,K;QACI,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UAC  
Z,IAAI,UAAU,oBAAV,CAAJ,C;YACI,IAAI,KAJ,C;cAAW,MAAM,8BAAyB,wDAAzB,C;YACjB,SAAS,O;YA  
CT,QAAQ,I;;QAGhB,IAAI,CAAC,KAAL,C;UAAy,MAAM,gCAAuB,6DAAvB,C;QAEIB,OAAO,4E;O;KafX,C;  
IAkBA,oC;MAII,OAAW,qBAAU,CAAd,GAAiB,qBAAK,CAAL,CAAjB,GAA8B,I;K;0FAGzC,yB;MAAA,6C;M  
AAA,oC;MAAA,gC;MAAA,uC;QAMoB,Q;QAFhB,aAAoB,I;QACpB,YAAY,K;QACI,0B;QAAhB,OAAgB,cAA  
hB,C;UAAgB,oC;UACZ,IAAI,UAAU,oBAAV,CAAJ,C;YACI,IAAI,KAJ,C;cAAW,OAAO,I;YACIB,SAAS,O;Y  
ACT,QAAQ,I;;QAGhB,IAAI,CAAC,KAAL,C;UAAy,OAAO,I;QACnB,OAAO,M;O;KAdX,C;IAiBA,+B;MIBzRI  
,IAAI,EkBiSI,KAAK,CIBjST,CAAJ,C;QACI,ckBgSc,wD;QIB/Rd,MAAM,gCAAyB,OAAQ,WAAjC,C;OkBgSV,  
OAAO,8BAAc,eAAF,CAAE,EAAa,gBAAb,CAAd,EAAoC,gBAApC,C;K;IAGX,+B;MIBrSI,IAAI,EkB6SI,KAAK  
,CIB7ST,CAAJ,C;QACI,ckB4Sc,wD;QIB3Sd,MAAM,gCAAyB,OAAQ,WAAjC,C;OkB4SV,OLhH6E,oBKgH1D,e  
AAF,CAAE,EAAa,gBAAb,CLhH0D,C;K;IKmHjF,kC;MIBjTI,IAAI,EkByTI,KAAK,CIBzTT,CAAJ,C;QACI,ckB  
wTc,wD;QIBvTd,MAAM,gCAAyB,OAAQ,WAAjC,C;OkBwTV,OAAO,mBAAkB,gBAAZ,mBAAS,CAAT,IAAY  
,EAAc,CAAd,CAAI,C;K;IAGX,mC;MIB7TI,IAAI,EkBqUI,KAAK,CIBrUT,CAAJ,C;QACI,ckBoUc,wD;QIBnUd  
,MAAM,gCAAyB,OAAQ,WAAjC,C;OkBoUV,OAAO,mBAAkB,gBAAZ,mBAAS,CAAT,IAAY,EAAc,CAAd,C  
AAIB,C;K;2FAGX,yB;MAAA,uD;MAAA,oC;MAAA,uC;QAMI,iBAAc,wBAAd,WAA+B,CAA/B,U;UACI,IAAI,  
CAAC,UAAU,iCAAK,KAAL,EAAV,CAAL,C;YACI,OAAO,8BAAy,CAAZ,EAAe,QAAQ,CAAR,IAAf,C;QACf,  
OAAO,E;O;KATX,C;4FAYA,yB;MAAA,uD;MAAA,oC;MAAA,uC;QAMI,iBAAc,wBAAd,WAA+B,CAA/B,U;U  
ACI,IAAI,CAAC,UAAU,iCAAK,KAAL,EAAV,CAAL,C;YACI,OL5JoF,oBK4JnE,CL5JmE,Ek4JhE,QAAQ,CA  
AR,IL5JgE,C;WK6J5F,OAAO,E;O;KATX,C;oFAYA,yB;MAAA,mD;MAAA,oC;MAAA,uC;QAMuB,UAAL,MA  
AK,EAAL,MAAK,EAAL,M;QAAK,mBAAL,SAAK,C;QAAL,mB;QAAA,kB;QAAA,kB;QAAd,0D;UACI,IAAI,  
CAAC,UAAU,iCAAK,KAAL,EAAV,CAAL,C;YACI,OAAO,8BAAy,KAAZ,EAAmB,gBAAnB,C;QACf,OAAO,  
E;O;KATX,C;oFAYA,yB;MAAA,mD;MAAA,oC;MAAA,uC;QAMuB,UAAL,MAAK,EAAL,MAAK,EAAL,M;Q  
AAK,mBAAL,SAAK,C;QAAL,mB;QAAA,kB;QAAA,kB;QAAd,0D;UACI,IAAI,CAAC,UAAU,iCAAK,KAAL,E  
AAV,CAAL,C;YACI,OLvLqE,oBKuLpD,KLvLoD,C;WKwL7E,OAAO,E;O;KATX,C;8EAYA,yB;MAAA,yD;M  
AkFA,oC;MAIFA,uC;QAMW,kBAAS,oB;QAKFM,Q;QAAA,uB;QAAtB,iBAAc,CAAd,wB;UACI,cAAc,qBAAI,  
KAJ,C;UACd,IApF6B,SAoFzB,CAAU,oBAAV,CAAJ,C;YAAwB,WAAy,gBAAO,OAAp,C;;QApFxC,OAsFO,

W;O;KA5FX,C;8EASA,yB;MAAA,yD;MAyEA,oC;MAzEA,uC;QAMW,kBAAS,oB;QAYEM,Q;QAAA,uB;QAAt  
B,iBAAC,CAAd,wB;UACI,cAAc,qBAAI,KAAJ,C;UACd,IA3E6B,SA2EzB,CAAU,oBAAV,CAAJ,C;YAAwB,WA  
AY,gBAAO,OAAP,C;;QA3ExC,OA6EO,WA7EqC,W;O;KANhD,C;4FASA,yB;MAAA,yD;MA+sBA  
,6C;MAAA,oC;MARuBA,uC;QAQW,kBAAGB,oB;QAouBV,gB;QADb,YAAY,C;QACC,0B;QAAb,OAAa,cAAb,  
C;UAAa,iC;UAAM,eAAO,cAAP,EAAO,sBAAP,S;UAAA,cAAGB,iB;UA7sB/B,IAvBoC,SAuBhC,CAAU,OA  
AAV,EAAiB,OAAjB,CAAJ,C;YAA2C,2BAAO,kBAAP,C;;QAvB/C,OAYBO,W;O;KAjCX,C;4FAWA,yB;MAAA,yD;  
MAWA,gC;MA+sBA,6C;MAAA,oC;MA1tBA,uC;QAQW,kBAAGB,oB;QAYtBV,gB;QADb,YAAY,C;QACC,0B;  
QAAb,OAAa,cAAb,C;UAAa,iC;UAAM,eAAO,cAAP,EAAO,sBAAP,S;UAAA,cAAGB,iB;UA7sB/B,IAZoC,SAY  
hC,CAAU,OA  
AAV,EAAiB,OAAjB,CAAJ,C;YAA2C,2BAAO,kBAAP,C;;QAZ/C,OAcO,WAd4C,W;O;KARvD,C;g  
GAWA,yB;MAAA,gC;MA+sBA,6C;MAAA,oC;MA/sBA,oD;QAstBiB,gB;QADb,YAAY,C;QACC,0B;QAAb,OA  
Aa,cAAb,C;UAAa,iC;UAAM,eAAO,cAAP,EAAO,sBAAP,S;UAAA,cAAGB,iB;UA7sB/B,IAAI,UAAU,OA  
AAV,E  
AAiB,OAAjB,CAAJ,C;YAA2C,2BAAO,kBAAP,C;;QAE/C,OAAO,W;O;KAXX,C;oFAcA,yB;MAAA,yD;MAk  
B  
A,6C;MAAA,oC;MAAA,gC;MA1BA,uC;QAMW,kBAAY,oB;QAKBH,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;U  
AAgB,oC;UAAM,IAAI,CAIBU,SakBT,CAAU,oBAAV,CAAL,C;YAAyB,WAAY,gBAAO,OAAP,C;;QAlB3D,O  
AmBO,W;O;KAZBX,C;oFASA,yB;MAAA,yD;MASA,6C;MAAA,oC;MAAA,gC;MATA,uC;QAMW,kBAAY,oB;  
QASH,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UAAM,IAAI,CATU,SAST,CAAU,oBAAV,CAAL,C  
;YAAyB,WAAY,gBAAO,OAAP,C;;QAT3D,OA  
UO,WAVwC,W;O;KANnD,C;wFASA,yB;MAAA,6C;MAAA,oC  
;MAAA,gC;MAAA,oD;QAMoB,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UAAM,IAAI,CAAC,UA  
A  
U,oBAAV,CAAL,C;YAAyB,WAAY,gBAAO,OAAP,C;;QAC3D,OAAO,W;O;KAPX,C;kFAUA,yB;MAAA,oC;M  
AA  
A,oD;QAM0B,Q;QAAA,uB;QAAtB,iBAAC,CAAd,wB;UACI,cAAc,qBAAI,KAAJ,C;UACd,IAAI,UAAU,oBA  
A  
V,CAAJ,C;YAAwB,WAAY,gBAAO,OAAP,C;;QAExC,OAAO,W;O;KAVX,C;IAaA,sC;MAII,IAAI,OAAQ,UA  
A  
Z,C;QAAuB,OAAO,E;MAC9B,OAAO,yBAAY,OAAZ,C;K;IAGX,sC;MAII,IAAI,OAAQ,UAAZ,C;QAAuB,OA  
A  
O,E;MAC9B,OAAO,uBAAU,OA  
AV,C;K;IAGX,sC;MAOc,Q;MAHV,WA  
AmB,wBAAR,OAAQ,EAAwB,EAAx  
B,C;MACnB,IAAI,SAAQ,CAAZ,C;QA  
Ae,OAAO,E;MACtB,aAAa,mBAAC,IAAd,C;MACH,yB;MAAV,OAAU,c  
A  
AV,C;QAAU,mB;QACN,MAAO,gBAAO,qBAAI,CAAJ,CAAP,C;;MAEX,OAAO,M;K;4EAGX,yB;MAAA,8B;  
MAAA,uC;MAAA,qC;QAKY,Q;QAAR,OAA8B,MAAtB,2DAAsB,EAAM,OAAN,CAAE,W;O;KALjD,C;IAQA,+  
B;MIB7fI,IAAI,EkBqgBI,KAAK,CIBrgBT,CAAJ,C;QACI,ckBogBc,wD;QIBngBd,MAAM,gCAAYB,OAAQ,WA  
A  
jC,C;OkBogBV,OAAO,8BAAY,CAAZ,EAAiB,eAAF,CAAE,EAAa,gBAAb,CAAjB,C;K;IAGX,+B;MIBzgBI,IA  
A  
I,EkBihBI,KAAK,CIBjhBT,CAAJ,C;QACI,ckBghBc,wD;QIB/gBd,MAAM,gCAAYB,OAAQ,WAAjC,C;OkBgh  
B  
V,OLjV4F,oBKiV3E,CLjV2E,EkIvtE,eAAF,CAAE,EAAa,gBAAb,CLjVsE,C;K;IKoVhG,kC;MIBrhBI,IAAI,Ek  
B  
6hBI,KAAK,CIB7hBT,CAAJ,C;QACI,ckB4hBc,wD;QIB3hBd,MAAM,gCAAYB,OAAQ,WAAjC,C;OkB4hBV,a  
A  
Aa,gB;MACb,OAAO,8BAAY,SAAW,eAAF,CAAE,EAAa,MAAb,CAAX,IAAZ,EAA6C,MAA7C,C;K;IAGX,m  
C;MIBliBI,IAAI,EkB0iBI,KAAK,CIB1iBT,CAAJ,C;QACI,ckByiBc,wD;QIBxiBd,MAAM,gCAAYB,OAAQ,WAA  
j  
C,C;OkByiBV,aAAa,gB;MACb,OL9W6E,oBK8W5D,SAAW,eAAF,CAAE,EAAa,MAAb,CAAX,IL9W4D,C;K;2  
F  
KiXjF,yB;MAAA,uD;MAAA,oC;MAAA,uC;QAMI,iBAAC,wBAAd,WAA+B,CAA/B,U;UACI,IAAI,CAAC,UA  
A  
U,iCAAK,KAAL,EAAV,CAAL,C;YACI,OAAO,8BAAY,QAAQ,CAAR,IAAZ,EAAuB,gBAAvB,C;;QAGf,OA  
A  
O,8BAAY,CAAZ,EAAe,gBAAf,C;O;KAXX,C;4FAcA,yB;MAAA,uD;MAAA,oC;MAAA,uC;QAMI,iBAAC,wB  
A  
Ad,WAA+B,CAA/B,U;UACI,IAAI,CAAC,UAAU,iCAAK,KAAL,EAAV,CAAL,C;YACI,OLvYqE,oBKuYpD,  
QAAQ,CAAR,ILvYoD,C;;QK0Y7E,OAAO,S;O;KAXX,C;oFAcA,yB;MAAA,oC;MAAA,uC;QAM0B,Q;QAAA,u  
B;QAAtB,iBAAC,CAAd,wB;UACI,IAAI,CAAC,UAAU,iCAAI,KAAJ,EAAV,CAAL,C;YACI,OAAO,8BAAY,CA  
A  
AZ,EAAe,KAAf,C;WAEf,OAAO,8BAAY,CAAZ,EAAe,gBAAf,C;O;KAVX,C;oFAaA,yB;MAAA,oC;MAAA,uC;  
QAM0B,Q;QAAA,uB;QAAtB,iBAAC,CAAd,wB;UACI,IAAI,CAAC,UAAU,iCAAI,KAAJ,EAAV,CAAL,C;YACI  
,OL/ZoF,oBK+ZnE,CL/ZmE,EK+ZhE,KL/ZgE,C;WKia5F,OAAO,S;O;KAVX,C;IAaA,gC;MAII,OAAO,qBAAC,S  
A  
Ad,CAAoB,U;K;kFAG/B,yB;MAAA,8B;MAAA,6C;MAAA,4B;QAKY,Q;QAAR,OAA8B,SAAtB,2DAAsB,CA  
A  
AW,W;O;KAL7C,C;oFAQA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MA4EA,6C;MAAA,oC;MAAA,gC;MA5EA,u  
C;QAWI,eAAmC,cAApB,YAAY,gBAAZ,CAAoB,EAAC,EAAd,C;QAC5B,kBAAY,mBAAoB,QAApB,C;QAYEH  
,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UACZ,WA1E8C,SA0E/B,CAAU,oBAAV,C;UzB9EnB,wB  
A  
AI,IAAK,MAAT,EAAGB,IAAK,OAARb,C;;QyBIA,OA4EO,W;O;KAXFX,C;wFAeA,yB;MAAA,0D;MAAA,yD;

MAAA,uE;MA6BA,6C;MAAA,oC;MAAA,gC;MA7BA,yC;QAWI,eAAmC,cAApB,YAAY,gBAAZ,CAAoB,EAA  
c,EAAd,C;QAC5B,kBAAc,mBAAuB,QAavB,C;QA2BL,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;U  
ACZ,WAAy,aA5BuC,WA4BnC,CAAY,oBAAZ,CAAJ,EAA0B,oBAA1B,C;;QA5BhB,OA8BO,W;O;KA1CX,C;w  
FAeA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MA8BA,6C;MAAA,oC;MAAA,gC;MA9BA,yD;QAUl,eAAmC,cAA  
pB,YAAY,gBAAZ,CAAoB,EAAC,EAAd,C;QAC5B,kBAAc,mBAAoB,QAAPB,C;QA6BL,Q;QAAA,0B;QAAhB,  
OAAgB,cAAhB,C;UAAgB,oC;UACZ,WAAy,aA9BoC,WA8BhC,CAAY,oBAAZ,CAAJ,EA9BiD,cA8BvB,CAAe  
,oBAAf,CAA1B,C;;QA9BhB,OAgCO,W;O;KA3CX,C;4FAcA,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,sD;  
QAUoB,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UACZ,WAAy,aAAI,YAAY,oBAAZ,CAAJ,EAA0  
B,oBAA1B,C;;QAEhB,OAAO,W;O;KAbX,C;4FAGBA,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,sE;QAUoB,  
Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UACZ,WAAy,aAAI,YAAY,oBAAZ,CAAJ,EAA0B,eAAe,o  
BAAf,CAA1B,C;;QAEhB,OAAO,W;O;KAbX,C;wFAGBA,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,oD;QAS  
oB,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UACZ,WAAe,UAAU,oBAAV,C;UzB9EnB,wBAAI,IAA  
K,MAAT,EAAGB,IAAK,OAAR,C;;QyBgFA,OAAO,W;O;KAZX,C;4FAeA,yB;MAAA,uD;MAAA,0D;MAAA,y  
D;MAAA,uE;MAgBA,6C;MAAA,oC;MAAA,gC;MAhBA,2C;QAYI,aAAa,mBAA6D,cAAtC,YAAmB,aAAP,gB  
AAO,EAAa,GAAb,CAAnB,CAAsC,EAAC,EAAd,CAA7D,C;QAcG,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UA  
AgB,oC;UAbO,MAcP,aAAI,oBAAJ,EAd,eAcF,CAAc,oBAAd,CAAb,C;;QAdhB,OAAuB,M;O;KAb3B,C;+FAGB  
A,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,wD;QAUoB,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,o  
C;UACZ,WAAy,aAAI,oBAAJ,EAAa,cAAc,oBAAd,CAAb,C;;QAEhB,OAAO,W;O;KAbX,C;IAGBA,iD;MALiB,  
Q;MAAA,4B;MAAb,OAAa,cAAb,C;QAAa,iC;QACT,WAAy,WAAl,iBAAJ,C;;MAEhB,OAAO,W;K;IAGX,iC;M  
AII,OAAO,2BAAa,eAAc,YAAmB,eAAP,gBAAO,EAAa,GAAb,CAAnB,CAAd,CAAb,C;K;IAGX,8B;MALiB,IAA  
N,I;MAAA,QAAM,gBAAN,C;aACH,C;UAAK,kB;UAAL,K;aACA,C;UAAK,cAAO,iCAAK,CAAL,EAAP,C;UA  
AL,K;gBACa,wBAAL,SAAK,C;UAHV,K;;MAAP,W;K;IAOJ,qC;MAII,OAAO,2BAAa,iBAAGB,gBAAhB,CAAb,  
C;K;IAGX,6B;MAMiB,IAAN,I;MAAA,QAAM,gBAAN,C;aACH,C;UAAK,iB;UAAL,K;aACA,C;UAAK,aAAM,i  
CAAK,CAAL,EAAN,C;UAAL,K;gBACQ,kCAAa,qBAAoB,YAAmB,eAAP,gBAAO,EAAa,GAAb,CAAnB,CAA  
pB,CAAb,C;UAHL,K;;MAAP,W;K;gFAOJ,yB;MAAA,+D;MA0CA,6C;MAAA,oC;MAAA,gD;MAAA,gC;MA1C  
A,uC;QAMW,kBAAU,gB;QAwCD,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UACZ,WAZC6B,SAyCl  
B,CAAU,oBAAV,C;UACC,OAAZ,WAAy,EAAO,IAAP,C;;QA1ChB,OA4CO,W;O;KAIDX,C;8FASA,yB;MAAA  
,+D;MAeA,6C;MAAA,oC;MAAA,gD;MAAA,gC;MAfA,uC;QAYW,kBAAiB,gB;QAcR,gB;QADhB,YAAY,C;Q  
ACI,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UACZ,WafoC,SAezB,EAAU,cAAV,EAAU,sBAAV,WAAmB,o  
BAAAnB,C;UACC,OAAZ,WAAy,EAAO,IAAP,C;;QAhBhB,OAkBO,W;O;KA9BX,C;kgAeA,yB;MAAA,6C;MA  
AA,oC;MAAA,gD;MAAA,gC;MAAA,oD;QAWoB,UACS,M;QAFzB,YAAY,C;QACI,0B;QAAhB,OAAgB,cAAh  
B,C;UAAgB,oC;UACZ,WAAW,WAAU,cAAV,EAAU,sBAAV,WAAmB,oBAAAnB,C;UACC,OAAZ,WAAy,EAA  
O,IAAP,C;;QAEhB,OAAO,W;O;KafX,C;oFAkBA,yB;MAAA,6C;MAAA,oC;MAAA,gD;MAAA,gC;MAAA,oD;  
QAIoB,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UACZ,WAAW,UAAU,oBAAV,C;UACC,OAAZ,W  
AAy,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KARX,C;gFAWA,yB;MAAA,wE;MAyBA,6C;MAAA,oC;MAAA,+D  
;MAAA,gC;MAzBA,yC;QASW,kBAAU,oB;QAYBD,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UACZ  
,UA1BoD,WA0B1C,CAAY,oBAAZ,C;UzBrjBP,U;UADP,YyBujBe,WzBvjBH,WyBujBwB,GzBvjBxB,C;UACL,I  
AAI,aAAJ,C;YACH,ayBqjBuC,gB;YAA5B,WzBpjBX,ayBojBgC,GzBpjBhC,EAAS,MAAT,C;YACA,e;;YAEA,c;  
;UyBijBA,iB;UACA,IAAK,WAAl,oBAAJ,C;;QA5BT,OA8BO,W;O;KAvcX,C;gFAYA,yB;MAAA,wE;MA8BA,6  
C;MAAA,oC;MAAA,+D;MAAA,gC;MA9BA,yD;QAUW,kBAAU,oB;QA8BD,Q;QAAA,0B;QAAhB,OAAgB,cA  
AhB,C;UAAgB,oC;UACZ,UA/BiD,WA+BvC,CAAY,oBAAZ,C;UzBvkBP,U;UADP,YyBykBe,WzBzkBH,WyBy  
kBwB,GzBzkBxB,C;UACL,IAAI,aAAJ,C;YACH,ayBukBuC,gB;YAA5B,WzBtkBX,ayBskBgC,GzBtkBhC,EAAS  
,MAAT,C;YACA,e;;YAEA,c;;UyBmkBA,iB;UACA,IAAK,WAjCyD,cAiCrD,CAAe,oBAAf,CAAJ,C;;QAJCT,OA  
mCO,W;O;KA7CX,C;oFAaA,yB;MAAA,6C;MAAA,oC;MAAA,+D;MAAA,gC;MAAA,sD;QASoB,Q;QAAA,0B;  
QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UACZ,UAAU,YAAY,oBAAZ,C;UzBrjBP,U;UADP,YyBujBe,WzBvjBH,  
WyBujBwB,GzBvjBxB,C;UACL,IAAI,aAAJ,C;YACH,ayBqjBuC,gB;YAA5B,WzBpjBX,ayBojBgC,GzBpjBhC,E  
AAS,MAAT,C;YACA,e;;YAEA,c;;UyBijBA,iB;UACA,IAAK,WAAl,oBAAJ,C;;QAEt,OAAO,W;O;KAdX,C;oF  
AiBA,yB;MAAA,6C;MAAA,oC;MAAA,+D;MAAA,gC;MAAA,sE;QAUoB,Q;QAAA,0B;QAAhB,OAAgB,cAAh

B,C;UAAgB,oC;UACZ,UAAU,YAAY,oBAAZ,C;UzBvkBP,U;UADP,YyBykBe,WzBzkBH,WyBykBwB,GzBzkBxB,C;UACL,IAAI,aAAJ,C;YACH,ayBukBuC,gB;YAA5B,WzBtkBX,ayBskBgC,GzBtkBhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;UyBmkBA,iB;UACA,IAAK,WAAI,eAAe,oBAAf,CAAJ,C;;QAET,OAAO,W;O;KafX,C;qFAkB A,yB;MAAA,6C;MAAA,oC;MAAA,kC;MAAA,4C;MAAA,wE;QAQW,sC;QAAA,8C;O;MARX,oDASQ,Y;QAAgD,OAAgB,SAAhB,oBAAgB,C;O;MATxE,iDAUQ,mB;QAAuC,gCAAY,oBAAZ,C;O;MAV/C,gF;MAAA,yC;QAQI,2D;O;KARJ,C;wEAcA,yB;MAAA,gE;MAYEA,6C;MAAA,oC;MAAA,gC;MAZEA,uC;QAOW,kBAAM,eAAa,gBAAb,C;QAuEA,Q;QAAA,0B;QAAb,OAAa,cAAb,C;UAAa,iC;UACT,WAA Y,WAXEmB,SAwEf,CAAU,iBA AV,CAAJ,C;;QAxEhB,OAYEO,W;O;KAhFX,C;sFAUA,yB;MAAA,gE;MA+BA,6C;MAAA,oC;MAAA,gC;MA/B A,uC;QAOW,kBAAa,eAAa,gBAAb,C;QA gCP,gB;QADb,YAAY,C;QACC,0B;QAAb,OAAa,cAAb,C;UAAa,iC;U ACT,WAA Y,WAJC0B,SAiCtB,EAAU,cAAV,EAAU,sBAAV,WAAmB,iBAAnB,CAAJ,C;;QAJChB,OAKCO,W;O ;KAZCX,C;mGAUA,yB;MAAA,+D;MAUA,gC;MAoLA,6C;MAAA,oC;MA9LA,uC;QAOW,kBAAoB,gB;QA8Ld ,gB;QADb,YAAY,C;QACC,0B;QAAb,OAAa,cAAb,C;UAAa,iC;UApLsB,U;UAAA,cAVQ,SAUR,EAoLT,cApLS, EAoLT,sBApLS,WAO LA,iBApLA,W;YAA6C,6B;;QAVhF,OAWO,W;O;KAIBX,C;uGAUA,yB;MAAA,gC;MAo LA,6C;MAAA,oC;MApLA,oD;QA2LiB,gB;QADb,YAAY,C;QACC,0B;QAAb,OAAa,cAAb,C;UAAa,iC;UApLsB ,U;UAAA,yBAoLT,cApLS,EAoLT,sBApLS,WAO LA,iBApLA,W;YAA6C,6B;;QACHF,OAAO,W;O;KARX,C;0F AWA,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,oD;QAQiB,UACiB,M;QAF9B,YAAY,C;QACC,0B;QAAb,O AAa,cAAb,C;UAAa,iC;UACT,WAA Y,WAAI,WAAU,cAAV,EAAU,sBAAV,WAAmB,iBAAnB,CAAJ,C;;QACH B,OAAO,W;O;KAVX,C;qFAaA,yB;MAAA,+D;MAUA,gC;MA2IA,6C;MAAA,oC;MArJA,uC;QAOW,kBAAa,g B;QAKJJ,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UA1IK,U;UAAA,cARe,SAQf,CA0IQ,oBA1IR,W; YAA sC,6B;;QAR3D,OASO,W;O;KAhBX,C;yFAUA,yB;MAAA,gC;MA2IA,6C;MAAA,oC;MA3IA,oD;QA+IoB, Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UA1IK,U;UAAA,wBA0IQ,oBA1IR,W;YAA sC,6B;;QAC3D ,OAAO,W;O;KANX,C;4EASA,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,oD;QAKiB,Q;QAAA,0B;QAAb,O AAa,cAAb,C;UAAa,iC;UACT,WAA Y,WAAI,UAAU,iBAAV,CAAJ,C;;QACHB,OAAO,W;O;KAPX,C;IAe4B,4C; MAAA,mB;QAAE,iC;O;K;IAL9B,iC;MAKI,OAAO,qBAAiB,6BAAjB,C;K;wEAGX,yB;MAAA,6C;MAAA,oC;M AAA,gC;MAAA,uC;QAMoB,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UAAM,IAAI,CAAC,UAAU,o BAAV,CAAL,C;YAAyB,OAAO,K;;QACtD,OAAO,I;O;KAPX,C;IAUA,2B;MAMI,OAAO,ECrwByC,qBAAU,C DqwBnD,C;K;wEAGX,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,uC;QAMoB,Q;QAAA,0B;QAAhB,OAAgB, cAAhB,C;UAAgB,oC;UAAM,IAAI,UAAU,oBAAV,CAAJ,C;YAAwB,OAAO,I;;QACrD,OAAO,K;O;KAPX,C;4E AUA,qB;MAKI,OAAO,gB;K;4EAGX,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,uC;QAKoB,Q;QADhB,YAA Y,C;QACI,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UAAM,IAAI,UAAU,oBAAV,CAAJ,C;YAAwB,qB;;QAC9 C,OAAO,K;O;KANX,C;0EASA,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,gD;QAUoB,Q;QADhB,kBAAkB, O;QACF,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UAAM,cAAc,UAAU,WAAV,EAAuB,oBAAvB,C;;QACpC, OAAO,W;O;KAXX,C;wFAcA,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,gD;QAYoB,UAA8B,M;QAF9C,YA AY,C;QACZ,kBAAkB,O;QACF,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UAAM,cAAc,WAAU,cAAV,EAAU, sBAAV,WAAmB,WAA nB,EAAGC,oBAAhC,C;;QACpC,OAAO,W;O;KAbX,C;mFAGBA,yB;MAAA,uD;MAAA, oC;MAAA,gD;QAYoC,Q;QAHhC,YAAY,wB;QACZ,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc, UAAU,kCAAI,YAAJ,EAAI,oBAAJ,SAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KAdX,C;iGaiBA,yB;MAA A,uD;MAAA,oC;MAAA,gD;QAUI,YAAY,wB;QACZ,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc, UAAU,KA AV,EAAiB,iCAAI,KA AJ,EAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAhBX,C;gFAM BA,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,oC;QAIoB,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,o C;UAAM,OAAO,oBAAP,C;;O;KAJ1B,C;8FAOA,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,oC;QAOiB,UAA a,M;QAD1B,YAAY,C;QACC,0B;QAAb,OAAa,cAAb,C;UAAa,iC;UAAM,QAAO,cAAP,EAAO,sBAAP,WAAgB, iBAAhB,C;;O;KAPvB,C;IAUA,2B;MAGI,OAAO,uB;K;4EAGX,yB;MAMA,uD;MAAA,oC;MANA,sC;QAGW,s B;;UAUP,ICz4BgD,qBAAU,CDy4B1D,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,qBAAK,CAAL,C;UACd,gBA AqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eAdmB,QAcJ,CAAS,oBAAT,C;U ACf,aAAU,cAAV,OAAa,SAAb,M;YACI,QAAQ,qBAAK,CAAL,C;YACR,QAJBe,QAiBP,CAAS,cAAT,C;YACR ,IAAI,2BAAW,CAAX,KA AJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QAvBP,yB;O;KAHJ,C;w FAMA,yB;MAAA,uD;MAAA,oC;MAAA,sC;QAOI,ICz4BgD,qBAAU,CDy4B1D,C;UAAe,OAAO,I;QACTB,cAA

c,qBAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eA Ae,SAAS,oBAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,qBAAK,CAAL,C;UACR,QAAQ,SAAS, cAAT,C;UACR,IAAI,2BAAW,CAAX,KAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KApBX, C;4EAuBA,yB;MAAA,sE;MAAA,oC;MAAA,uD;MdznCA,iB;McyCA,sC;QAeiB,Q;QAFb,ICt6BgD,qBAAU,CD s6B1D,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,iCAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UA CI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,WdloCG,MAAO,KckoCO,QdloCP,EckoCiB,CdloCjB,C;;QcooC d,OAAO,Q;O;KAnBX,C;4EAsBA,yB;MAAA,sE;MAAA,oC;MAAA,uD;Md1pCA,iB;McOpCA,sC;QAeiB,Q;QAF b,IC57BgD,qBAAU,CD47B1D,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,iCAAK,CAAL,EAAT,C;QACF,+B;QA Ab,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,WdnqCG,MAAO,KcmqCO,QdnqCP,Ec mqCiB,CdnqCjB,C;;QcqqCd,OAAO,Q;O;KAnBX,C;4EAsBA,yB;MAAA,sE;MAAA,oC;MAAA,uD;MAAA,sC;Q AaiB,Q;QAFb,IC9BgD,qBAAU,CDg9B1D,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,iCAAK,CAAL,EAAT,C;Q ACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,IAAI,2BAAW,CAAX,KA AJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;wFAsBA,yB;MAAA,oC;MAAA,uD;Md3rCA,iB;Mc2rCA ,sC;QAaiB,Q;QAFb,ICt+Bgd,qBAAU,CDs+B1D,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,iCAAK,CAAL,EAAT,C ;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,WdlsCG,MAAO,KcksC O,QdlsCP,EcksCiB,CdlsCjB,C;;QcosCd,OAAO,Q;O;KAjBX,C;wFAoBA,yB;MAAA,oC;MAAA,uD;Md1tCA,iB; Mc0tCA,sC;QAaiB,Q;QAFb,IC1/BgD,qBAAU,CD0/B1D,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,iCAAK,CAAL, EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,WdjuCG,MAA O,KciuCO,QdjuCP,EciuCiB,CdjuCjB,C;;QcmuCd,OAAO,Q;O;KAjBX,C;wFAoBA,yB;MAAA,oC;MAAA,uD;M AAA,sC;QAWiB,Q;QAFb,IC5gCgD,qBAAU,CD4gC1D,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,iCAAK,CAAL,E AAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,IAAI,2BAAW,C AAX,KAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;oFAoBA,yB;MAAA,sE;MAAA,oC;MAAA,uD; MAAA,kD;QAaiB,Q;QAFb,ICliCgD,qBAAU,CDkiC1D,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,iCAAK,CAAL ,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,IAAI,UAAW,S AAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;gGAs BA,yB;MAAA,oC;MAAA,uD;MAAA,kD;QAWiB,Q;QAFb,ICtjCgD,qBAAU,CDsjC1D,C;UAAe,OAAO,I;QACt B,eAAe,SAAS,iCAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,E AAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAAkC,CAAtC,C;YACI,WAAW,C;;QAGnB ,OAAO,Q;O;KAjBX,C;IAoBA,iC;MAOiB,Q;MAFb,ICtkCgD,qBAAU,CDskC1D,C;QAAe,OAAO,I;MACtB,UAA U,qBAAK,CAAL,C;MACG,kC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,qBAAK,CAAL,C;QACR,IAAI,MAAM,C AAV,C;UAAa,MAAM,C;;MAEvB,OAAO,G;K;IAGX,2C;MAGI,OAAO,4BAAc,UAAAd,C;K;IAGX,iD;MAOiB,Q; MAFb,IC11CgD,qBAAU,CD01C1D,C;QAAe,OAAO,I;MACtB,UAAU,qBAAK,CAAL,C;MACG,kC;MAAb,aAAU ,CAAV,iB;QACI,QAAQ,qBAAK,CAAL,C;QACR,IAAI,UAAW,SAAQ,gBAAR,EAAa,cAAb,CAAX,GAA6B,CA AjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IAGX,2B;MAGI,OAAO,uB;K;4EAGX,yB;MAMA,uD;MAAA,o C;MANA,sC;QAGW,sB;;UAUP,ICtnCgD,qBAAU,CDsnC1D,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,qBAAK ,CAAL,C;UACd,gBAAqB,wB;UACrB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eAdmB,QA cJ,CAAS,oBAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,qBAAK,CAAL,C;YACR,QAjBe,QAiBP, CAAS,cAAT,C;YACR,IAAI,2BAAW,CAAX,KAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;Q AvBP,yB;O;KAHJ,C;wFAMA,yB;MAAA,uD;MAAA,oC;MAAA,sC;QAoi,ICtnCgD,qBAAU,CDsnC1D,C;UAAe ,OAAO,I;QACtB,cAAc,qBAAK,CAAL,C;QACd,gBAAqB,cAAL,SAAK,C;QACrB,IAAI,cAAa,CAAjB,C;UAAo B,OAAO,O;QAC3B,eAAe,SAAS,oBAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,qBAAK,CAAL, C;UACR,QAAQ,SAAS,cAAT,C;UACR,IAAI,2BAAW,CAAX,KAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAG nB,OAAO,O;O;KApBX,C;4EAuBA,yB;MAAA,sE;MAAA,oC;MAAA,uD;Md1pCA,iB;MckpCA,sC;QAeiB,Q;QA Fb,ICnpCgD,qBAAU,CDmpC1D,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,iCAAK,CAAL,EAAT,C;QACF,+B;Q AAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,Wd3pCG,MAAO,Kc2pCO,Qd3pCP,E c2pCiB,Cd3pCjB,C;;Qc6pCd,OAAO,Q;O;KAnBX,C;4EAsBA,yB;MAAA,sE;MAAA,oC;MAAA,uD;MdnrCA,iB; McmrCA,sC;QAeiB,Q;QAFb,ICzqCgD,qBAAU,CDyqC1D,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,iCAAK,CA AL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,Wd5rCG,M

AAO,Kc4rCO,Qd5rCP,Ec4rCiB,Cd5rCjB,C;;Qc8rCd,OAAO,Q;O;KAnBX,C;4EAsBA,yB;MAAA,sE;MAAA,oC;MAAA,uD;MAAA,sC;QAaiB,Q;QAFb,IC7rCgD,qBAAU,CD6rC1D,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,iCAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;wFAsBA,yB;MAAA,oC;MAAA,uD;MdptCA,iB;McotCA,sC;QAaiB,Q;QAFb,ICntCgD,qBAAU,CDmtC1D,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,iCAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,Wd3tCG,MAAO,Kc2tCO,Qd3tCP,Ec2tCiB,Cd3tCjB,C;;Qc6tCd,OAAO,Q;O;KAjBX,C;wFAoBA,yB;MAAA,oC;MAAA,uD;MdnvCA,iB;McmvCA,sC;QAaiB,Q;QAFb,ICvuCgD,qBAAU,CDuuC1D,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,iCAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,Wd1vCG,MAAO,Kc0vCO,Qd1vCP,Ec0vCiB,Cd1vCjB,C;;Qc4vCd,OAAO,Q;O;KAjBX,C;wFAoBA,yB;MAAA,oC;MAAA,uD;MAAA,sC;QAWiB,Q;QAFb,ICzvCgD,qBAAU,CDyvC1D,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,iCAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;oFAoBA,yB;MAAA,sE;MAAA,oC;MAAA,uD;MAAA,kD;QAaiB,Q;QAFb,IC/wCgD,qBAAU,CD+wC1D,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,iCAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAnBX,C;gGAsBA,yB;MAAA,oC;MAAA,uD;MAAA,kD;QAWiB,Q;QAFb,ICnyCgD,qBAAU,CDmyC1D,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,iCAAK,CAAL,EAAT,C;QACF,+B;QAAb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,iCAAK,CAAL,EAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAjBX,C;IAoBA,iC;MAOiB,Q;MAFb,ICnzCgD,qBAAU,CDmzC1D,C;QAAe,OAAO,I;MACtB,UAAU,qBAAK,CAAL,C;MACG,kC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,qBAAK,CAAL,C;QACR,IAAI,MAAM,CAAV,C;UAAa,MAAM,C;;MAEvB,OAAO,G;K;IAGX,2C;MAGI,OAAO,4BAAc,UAAAd,C;K;IAGX,iD;MAOiB,Q;MAFb,ICv0CgD,qBAAU,CDu0C1D,C;QAAe,OAAO,I;MACtB,UAAU,qBAAK,CAAL,C;MACG,kC;MAAb,aAAU,CAAV,iB;QACI,QAAQ,qBAAK,CAAL,C;QACR,IAAI,UAAW,SAAQ,gBAAR,EAAa,cAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IAGX,4B;MAMIL,OCt1CgD,qBAAU,C;K;0EDy1C9D,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,uC;QAMoB,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UAAM,IAAI,UAAU,oBAAV,CAAJ,C;YAAwB,OAAO,K;;QACrD,OAAO,I;O;KAPX,C;8EAUAY,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,oC;QAKmC,Q;QAAA,0B;QAAhB,OAAgB,cAAhB,C;UAAgB,oC;UAAM,OAAO,oBAAP,C;;QAARc,gB;O;KALJ,C;4FAQA,yB;MAAA,6B;MAAA,sC;MA/fA,6C;MAAA,oC;MAAA,gC;MA+FA,2BAQiB,yB;QAvGbjB,6C;QAAA,oC;QAAA,gC;eAugBiB,0B;UAAA,4B;YAAE,aAAe,c;YAhgBjB,gB;YADb,YAAY,C;YACC,0B;YAAb,OAAa,cAAb,C;CAAA,iC;CAAM,QAAO,cAAP,EAAO,sBAAP,WAAgB,iBAAhB,C;;YAggBmB,W;W;S;OAAzB,C;MARjB,oC;QAXfiB,gB;QADb,YAAY,C;QACC,0B;QAAb,OAAa,cAAb,C;UAAa,iC;UAAM,QAAO,cAAP,EAAO,sBAAP,WAAgB,iBAAhB,C;;QAggBnB,gB;O;KARJ,C;8EAWA,yB;MAAA,4F;MAAA,uD;MAAA,oC;MAAA,gC;MAAA,uC;QAgBqB,Q;QAHjB,ICz5CgD,qBAAU,CDy5C1D,C;UACI,MAAM,mCAA8B,uCAA9B,C;QACV,kBAakB,qBAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,oBAAU,wBAAV,EAAuB,iCAAK,KAAL,EAAvB,E;;QAEIB,OAAO,W;O;KAnBX,C;4FAsBA,yB;MAAA,4F;MAAA,uD;MAAA,oC;MAAA,gC;MAAA,uC;QAgBqB,Q;QAHjB,ICz5CgD,qBAAU,CDy5C1D,C;UACI,MAAM,mCAA8B,uCAA9B,C;QACV,kBAakB,qBAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,oBAAU,KAAY,EAAiB,wBAAjB,EAA8B,iCAAK,KAAL,EAA9B,E;;QAEIB,OAAO,W;O;KAnBX,C;wGAsBA,yB;MAAA,uD;MAAA,oC;MAAA,gC;MAAA,uC;QAgBqB,Q;QAHjB,IC/6CgD,qBAAU,CD+6C1D,C;UACI,OAAO,I;QACX,kBAakB,qBAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,oBAAU,KAAY,EAAiB,wBAAjB,EAA8B,iCAAK,KAAL,EAA9B,E;;QAEIB,OAAO,W;O;KAnBX,C;0FAsBA,yB;MAAA,uD;MAAA,oC;MAAA,gC;MAAA,uC;QAIbqB,Q;QAHjB,ICt8CgD,qBAAU,CDs8C1D,C;UACI,OAAO,I;QACX,kBAakB,qBAAK,CAAL,C;QACD,+B;QAAjB,iBAAc,CAAd,yB;UACI,cAAc,oBAAU,wBAAV,EAAuB,iCAAK,KAAL,EAAvB,E;;QAEIB,OAAO,W;O;KApBX,C;uFAuBA,yB;MAAA,uD;MAAA,4F;MAAA,oC;MAAA,gC;MAAA,uC;QAE0B,UAEU,M;QAJhC,YAAY,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,uCAA9B,C;QACrB,kBAakB,sBAAI,YAAJ,EAAI,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,oBAAU,kCAAI,cAAJ,EAAI,sBAAJ,WAAV,EAAwB,wBAAxB,E;;QAEIB,OAAO,W;O;KAnBX,C;qGAsBA,yB;MAAA,uD;MAAA,4F;MAAA,oC;

MAAA,gC;MAAA,uC;QAE0B,Q;QAFtB,YAA Y,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,uCA  
A9B,C;QACrB,kBAAkB,sBAAI,YAAJ,EAAI,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,oBAAU,K  
AAV,EAAiB,iCAAI,KAAJ,EAAjB,EAA6B,wBAA7B,E;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;iHAuBA,yB;M  
AAA,uD;MAAA,oC;MAAA,gC;MAAA,uC;QAE0B,Q;QAFtB,YAA Y,wB;QACZ,IAAI,QAAQ,CAAZ,C;UAAe,O  
AAO,I;QACtB,kBAAkB,sBAAI,YAAJ,EAAI,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,oBAAU,K  
AAV,EAAiB,iCAAI,KAAJ,EAAjB,EAA6B,wBAA7B,E;UACd,qB;;QAEJ,OAAO,W;O;KApBX,C;mGAuBA,yB;  
MAAA,uD;MAAA,oC;MAAA,gC;MAAA,uC;QAgB0B,UAEU,M;QAJhC,YAA Y,wB;QACZ,IAAI,QAAQ,CAAZ,  
C;UAAe,OAAO,I;QACtB,kBAAkB,sBAAI,YAAJ,EAAI,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,  
oBAAU,kCAAI,cAAJ,EAAI,sBAAJ,WAAV,EAAwB,wBAAxB,E;;QAEIB,OAAO,W;O;KApBX,C;wFAuBA,yB;  
MAAA,gD;MAAA,gE;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,gD;QAgBoB,Q;QAHhB,ICvjDgD,qBAAU,CDu  
jD1D,C;UAAe,OAAO,OAAO,OAAP,C;QACgB,kBAAzB,eAAa,mBAAS,CAAT,IAAb,C;QAAiC,8B;QAA9C,af5  
wDO,W;Qe6wDP,kBAAkB,O;QACF,0B;QAAhB,OAAGB,cAAhB,C;UAAgB,oC;UACZ,cAAc,UAAU,WAAV,EA  
AuB,oBAAvB,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KApBX,C;sGAuBA,yB;MAAA,gD;MAAA  
,gE;MAAA,mD;MAAA,oC;MAAA,gD;QAIbKb,gC;QAHd,IC/kDgD,qBAAU,CD+kD1D,C;UAAe,OAAO,OAAO,  
OAAP,C;QACgB,kBAAzB,eAAa,mBAAS,CAAT,IAAb,C;QAAiC,8B;QAA9C,afpyDO,W;QeqyDP,kBAAkB,O;Q  
ACJ,6B;QAAA,mB;QAAA,kB;QAAA,kB;QAAAd,0D;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,iCAA  
K,KAAL,EAA9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KArBX,C;4FAwBA,yB;MAAA,qD;MA  
AA,gE;MAAA,oC;MAAA,gC;MAAA,uC;QAgB0B,Q;QAHtB,ICTmDgD,qBAAU,CDsmD1D,C;UAAe,OAAO,W;  
QACtB,sBAAkB,qBAAK,CAAL,CAAI,C;QACqC,kBAAXB,eAAGB,gBAAhB,C;QAAgC,sBAAI,0BAAJ,C;QA  
A7C,af5zDO,W;Qe6zDe,uB;QAAtB,iBAAC,CAAd,wB;UACI,gBAAC,oBAAU,0BAAV,EAAuB,iCAAK,KAAL,E  
AAvB,E;UACd,MAAO,WAAI,0BAAJ,C;;QAEX,OAAO,M;O;KApBX,C;0GAuBA,yB;MAAA,qD;MAAA,gE;M  
AAA,oC;MAAA,gC;MAAA,uC;QAIb0B,Q;QAHtB,IC9nDgD,qBAAU,CD8nD1D,C;UAAe,OAAO,W;QACtB,sB  
AAkB,qBAAK,CAAL,CAAI,C;QACqC,kBAAXB,eAAGB,gBAAhB,C;QAAgC,sBAAI,0BAAJ,C;QAA7C,afp1D  
O,W;Qeq1De,uB;QAAtB,iBAAC,CAAd,wB;UACI,gBAAC,oBAAU,KAAV,EAAiB,0BAAjB,EAA8B,iCAAK,KA  
AL,EAA9B,E;UACd,MAAO,WAAI,0BAAJ,C;;QAEX,OAAO,M;O;KArBX,C;0EAwBA,yB;MA9FA,gD;MAAA,  
gE;MAAA,6C;MAAA,oC;MAAA,gC;MA8FA,gD;QAcW,sB;;UA5FS,Q;UAHhB,ICvjDgD,qBAAU,CDujD1D,C;  
YAAe,qBAAO,OA+FH,OA/FG,C;YAAP,uB;WACuB,kBAAzB,eAAa,mBAAS,CAAT,IAAb,C;UAAiC,sBA8F3B,  
OA9F2B,C;UAA9C,af5wDO,W;Ue6wDP,kBA6FmB,O;UA5FH,0B;UAAhB,OAAGB,cAAhB,C;YAAgB,oC;YAC  
Z,cA2FwB,SA3FV,CAAU,WAAV,EAAuB,oBAAvB,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;;QA  
wFP,yB;O;KAdJ,C;wFAiBA,yB;MAxFA,gD;MAAA,gE;MAAA,mD;MAAA,oC;MAwFA,gD;QAEW,6B;;UatFO,  
gC;UAHd,IC/kDgD,qBAAU,CD+kD1D,C;YAAe,4BAAO,OAYFI,OAzFJ,C;YAAP,8B;WACuB,kBAAzB,eAAa,m  
BAAS,CAAT,IAAb,C;UAAiC,sBAwFpB,OAxFoB,C;UAA9C,afpyDO,W;UeqyDP,kBAuF0B,O;UatfZ,6B;UAA  
A,mB;UAAA,kB;UAAA,kB;UAAAd,0D;YACI,cAqF+B,SArFjB,CAAU,KAAV,EAAiB,WAAjB,EAA8B,iCAAK,K  
AAL,EAA9B,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,M;;;QAKFP,gC;O;KAFj,C;4EAKBA,yB;MAAA,6  
C;MAAA,oC;MAAA,gC;MAAA,sC;QAOoB,Q;QADhB,UAAe,C;QACC,0B;QAAhB,OAAGB,cAAhB,C;UAAgB,  
oC;UACZ,YAAO,SAAS,oBAAT,CAAP,I;;QAEJ,OAAO,G;O;KAVX,C;wFAAa,yB;MAAA,6C;MAAA,oC;MAA  
A,gC;MAAA,sC;QAOoB,Q;QADhB,UAAkB,G;QACF,0B;QAAhB,OAAGB,cAAhB,C;UAAgB,oC;UACZ,OAAO,  
SAAS,oBAAT,C;;QAEX,OAAO,G;O;KAVX,C;4EAAa,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,sC;QAUoB  
,Q;QADhB,UAAoB,C;QACJ,0B;QAAhB,OAAGB,cAAhB,C;UAAgB,oC;UACZ,OAAO,SAAS,oBAAT,C;;QAEX,  
OAAO,G;O;KAbX,C;4EAgBA,yB;MAAA,6C;MAAA,oC;MAAA,gC;MAAA,sC;QAUoB,Q;QADhB,UAAe,C;QA  
CC,0B;QAAhB,OAAGB,cAAhB,C;UAAgB,oC;UACZ,YAAO,SAAS,oBAAT,CAAP,I;;QAEJ,OAAO,G;O;KAbX,  
C;4EAgBA,yB;MAAA,SASoB,gB;MATpB,6C;MAAA,oC;MAAA,gC;MAAA,sC;QAUoB,Q;QADhB,Y;QACgB,0  
B;QAAhB,OAAGB,cAAhB,C;UAAgB,oC;UACZ,cAAO,SAAS,oBAAT,CAAP,C;;QAEJ,OAAO,G;O;KAbX,C;4E  
AgBA,yB;MAAA,6C;MAAA,oC;MAAA,gC;M7BppDA,6B;M6BopDA,sC;QAWoB,Q;QADhB,U7BppDmC,c6Bo  
pDnB,C7BppDmB,C;Q6BqpDnB,0B;QAAhB,OAAGB,cAAhB,C;UAAgB,oC;UACZ,M7Bx9DiD,c6Bw9DjD,G7B  
x9D2D,KAAX,G6Bw9DzD,SAAS,oBAAT,C7Bx9DoE,KAAX,IAAf,C;;Q6B09DrD,OAAO,G;O;KADx,C;4EAI  
A,yB;MAAA,6C;MAAA,oC;MAAA,gC;MblqDA,+B;MakqDA,sC;QAWoB,Q;QADhB,UbjqDqC,eAAW,oBaiqD/  
B,CbjqD+B,CAAX,C;QakqDrB,0B;QAAhB,OAAGB,cAAhB,C;UAAgB,oC;UACZ,Mbt+DmD,eas+DnD,Gbt+D8





M,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,O;;MACrD,OAAO,I;K;+FAGX,gC;MAOoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAGB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,O;;MACrD,OAAO,I;K;2FAGX,yB;MAkqGI,8D;MAIqGJ,iD;QAOe,oBAAS,C;QAAT,S;UAAc,gBA2pGT,cAAR,iBAAQ,C;SA3pGhB,OAAO,OAAcS,sBAAL,KAAJ,CAAtC,GAAsD,aAAa,KAAb,C;O;KAPjE,C;2FAUA,yB;MAGqGI,8D;MAhqGJ,iD;QAOe,oBAAS,C;QAAT,S;UAAc,gBAypGT,cAAR,iBAAQ,C;SAzpGhB,OAAO,OAAcS,sBAAL,KAAJ,CAAtC,GAAsD,aAAa,KAAb,C;O;KAPjE,C;2FAUA,yB;MA8pGI,8D;MA9pGJ,iD;QAOe,oBAAS,C;QAAT,S;UAAc,gBAupGT,cAAR,iBAAQ,C;SAvpGhB,OAAO,OAAcS,sBAAL,KAAJ,CAAtC,GAAsD,aAAa,KAAb,C;O;KAPjE,C;2FAUA,yB;MA4pGI,8D;MA5pGJ,iD;QAOe,oBAAS,C;QAAT,S;UAAc,gBAqpGT,cAAR,iBAAQ,C;SArpGhB,OAAO,OAAcS,sBAAL,KAAJ,CAAtC,GAAsD,aAAa,KAAb,C;O;KAPjE,C;IAUA,wC;MAQe,oBAAS,C;MAAT,S;QAAC,gBAknGT,gBAAR,iBAAQ,C;OAlnGhB,OAAO,OAAcS,sBAAL,KAAJ,CAAtC,GAAsD,I;K;IAGjE,wC;MAQe,oBAAS,C;MAAT,S;QAAC,gBA+mGT,gBAAR,iBAAQ,C;OA/mGhB,OAAO,OAAcS,sBAAL,KAAJ,CAAtC,GAAsD,I;K;IAGjE,wC;MAQe,oBAAS,C;MAAT,S;QAAC,gBA4mGT,gBAAR,iBAAQ,C;OA5mGhB,OAAO,OAAcS,sBAAL,KAAJ,CAAtC,GAAsD,I;K;IAGjE,wC;MAQe,oBAAS,C;MAAT,S;QAAC,gBAYmGT,gBAAR,iBAAQ,C;OAZmGhB,OAAO,OAAcS,sBAAL,KAAJ,CAAtC,GAAsD,I;K;uFAGjE,yB;MAAA,kD;MAAA,qC;QAOI,OAAe,QAAR,iBAAQ,EAAQ,OnCtdU,KmCsdIB,C;O;KAPnB,C;uFAUA,yB;MAAA,kD;MAAA,qC;QAOI,OAAe,QAAR,iBAAQ,EAAQ,OnBrdY,KmBqdpB,C;O;KAPnB,C;uFAUA,yB;MAAA,kD;MAAA,qC;QAOI,OAAe,QAAR,iBAAQ,EAAQ,OlCjhBc,KkCihBtB,C;O;KAPnB,C;iGAUA,yB;MAAA,sC;MnC5ZA,6B;MmC4ZA,0BAOGC,yB;QnCnahC,6B;emCmagC,6B;UAAA,qB;YAAE,yBnCzZK,cmCyZK,EnCzZL,CmCyZL,C;W;S;OAAF,C;MAPhC,uC;QAOMB,kBAAR,iB;QAAQ,uB;;UtC40Bf,0D;YACI,IsC70B0B,UnCzZK,cHsuCjB,YAAK,KAAL,CGtuCiB,CmCyZL,CtC60B1B,C;cACI,sBAAO,K;cAAP,wB;;UAGR,sBAAO,E;;;QsCj1BP,0B;O;KAPJ,C;iGAUA,yB;MAAA,sC;MnBvZA,+B;MmBuZA,0BAOGC,yB;QnB9ZhC,+B;emB8ZgC,6B;UAAA,qB;YAAE,yBnBpZQ,emBoZE,EnBpZF,CmBoZR,C;W;S;OAAF,C;MAPhC,uC;QAOMB,kBAAR,iB;QAAQ,uB;;UtC80Bf,0D;YACI,IsC/0B0B,UnBpZQ,enBmuCpB,YAAK,KAAL,CmBnuCoB,CmBoZR,CtC+0B1B,C;cACI,sBAAO,K;cAAP,wB;;UAGR,sBAAO,E;;;QsCn1BP,0B;O;KAPJ,C;iGAUA,yB;MAAA,sC;MpC9dA,+B;MoC8dA,0BAOGC,yB;QpCrehC,+B;eoCqegC,6B;UAAA,qB;YAAE,yBpC3dQ,eoC2dE,EpC3dF,CoC2dR,C;W;S;OAAF,C;MAPhC,uC;QAOMB,kBAAR,iB;QAAQ,uB;;UtCgyBf,0D;YACI,IsCjyB0B,UpC3dQ,eF4vCpB,YAAK,KAAL,CE5vCoB,CoC2dR,CtCiyB1B,C;cACI,sBAAO,K;cAAP,wB;;UAGR,sBAAO,E;;;QsCryBP,0B;O;KAPJ,C;iGAUA,yB;MAAA,sC;MIC3dA,iC;MkC2dA,0BAOGC,yB;QlClehC,iC;ekCkegC,6B;UAAA,qB;YAAE,yBICxdW,gBkCwdD,EICxdC,CkCwdX,C;W;S;OAAF,C;MAPhC,uC;QAOMB,kBAAR,iB;QAAQ,uB;;UtCkyBf,0D;YACI,IsCnyB0B,UICxdW,gBJ2vCvB,YAAK,KAAL,CI3vCuB,CkCwdX,CtCmyB1B,C;cACI,sBAAO,K;cAAP,wB;;UAGR,sBAAO,E;;;QsCvyBP,0B;O;KAPJ,C;+FAUA,yB;MAAA,sC;MtCm5BA,0D;MAAA,+C;MGv1CA,6B;MmCocA,yBAO+B,yB;QnC3c/B,6B;emC2c+B,6B;UAAA,qB;YAAE,yBnCjcm,cmCicI,EnCjCj,CmCicN,C;W;S;OAAF,C;MAP/B,uC;QAOMB,kBAAR,iB;QAAQ,sB;;UtCg5BD,Q;UAAA,OAAQ,SAAR,wBAAQ,CAAR,W;UAAAd,OAAc,cAAAd,C;YAAc,uB;YACV,IsCj5ByB,UnCjcm,CkHk1CjB,YAAK,KAAL,CGI1CiB,CmCicN,CtCi5BzB,C;cACI,qBAAO,K;cAAP,uB;;UAGR,qBAAO,E;;;QsCr5BP,yB;O;KAPJ,C;+FAUA,yB;MAAA,sC;MtCq5BA,0D;MAAA,+C;MmBp1CA,+B;MmB+bA,yBAO+B,yB;QnBtc/B,+B;emBsc+B,6B;UAAA,qB;YAAE,yBnB5bS,emB4bC,EnB5bD,CmB4bT,C;W;S;OAAF,C;MAP/B,uC;QAOMB,kBAAR,iB;QAAQ,sB;;UtCk5BD,Q;UAAA,OAAQ,SAAR,wBAAQ,CAAR,W;UAAAd,OAAc,cAAAd,C;YAAc,uB;YACV,IsCn5ByB,UnB5bS,enB+0CpB,YAAK,KAAL,CmB/0CoB,CmB4bT,CtCm5BzB,C;cACI,qBAAO,K;cAAP,uB;;UAGR,qBAAO,E;;;QsCv5BP,yB;O;KAPJ,C;+FAUA,yB;MAAA,sC;MtCu2BA,0D;MAAA,+C;ME72CA,+B;MoCsgBA,yBAO+B,yB;QpC7gB/B,+B;eoC6gB+B,6B;UAAA,qB;YAAE,yBpCngBS,eoCmgBC,EpCngBD,CoCmgBT,C;W;S;OAAF,C;MAP/B,uC;QAOMB,kBAAR,iB;QAAQ,sB;;UtCo2BD,Q;UAAA,OAAQ,SAAR,wBAAQ,CAAR,W;UAAAd,OAAc,cAAAd,C;YAAc,uB;YACV,IsCr2ByB,UpCngBS,eFw2CpB,YAAK,KAAL,CEx2CoB,CoCmgBT,CtCq2BzB,C;cACI,qBAAO,K;cAAP,uB;;UAGR,qBAAO,E;;;QsCz2BP,yB;O;KAPJ,C;+FAUA,yB;MAAA,sC;MtCy2BA,0D;MAAA,+C;MI52CA,iC;MkCmgBA,yBAO+B,yB;QIC1gB/B,iC;ekC0gB+B,6B;UAAA,qB;YAAE,yBIChgBY,gBkCggBF,ElChgBE,CkCggBZ,C;W;S;OAAF,C;MAP/B,uC;QAOMB,kBAAR,iB;QAAQ,sB;;UtCs2BD,Q;UAAA,OAAQ,SAAR,wBAAQ,CAAR,W;UAAAd,OAAc,cAAAd,C;YAAc,uB;YACV,IsCv2ByB,UIChgBY,gBJu2CvB,YAAK,KAAL,Civ2CuB,CkCggBZ,CtCu2BzB,C;cACI,qBAAO,K;cAAP,uB;;UAGR,qBAAO,E;;;QsC32BP,yB;O;KAPJ,C;iFAUA,yB;M

AAA,4C;MnC5eA,6B;MmC4eA,4B;QAWI,OnC7emC,cmC6epB,KAAR,iBAAQ,CnC7eoB,C;O;KmCkevC,C;iFAcA,yB;MAAA,4C;MnB3eA,+B;MmB2eA,4B;QAWI,OnB5esC,emB4evB,KAAR,iBAAQ,CnB5euB,C;O;KmBie1C,C;iFAcA,yB;MAAA,4C;MpCtjBA,+B;MoCsjBA,4B;QAWI,OpCvjBsC,eoCujBvB,KAAR,iBAAQ,CpCvjBuB,C;O;KoC4iB1C,C;iFAcA,yB;MAAA,4C;MlCvjBA,iC;MkCujBA,4B;QAWI,OiCxBjByC,gBkCwjB1B,KAAR,iBAAQ,CiCxB0B,C;O;KkC6iB7C,C;iFAcA,yB;MAAA,+C;MAAA,iE;MA83FI,0D;MA93FJ,uC;QAWkB,Q;QAAA,OAAa,SAm3FX,YAn3FF,SAm3FN,QAAQ,CAn3FW,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAAc,sBAAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,MAAM,gCAAuB,mDAAvB,C;O;KafV,C;iFAkBA,yB;MAAA,+C;MAAA,iE;MAo3FI,0D;MAp3FJ,uC;QAWkB,Q;QAAA,OAAa,SAy2FX,YAz2FF,SAy2FN,QAAQ,CAz2FW,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAAc,sBAAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,MAAM,gCAAuB,mDAAvB,C;O;KafV,C;iFAkBA,yB;MAAA,+C;MAAA,iE;MA02FI,0D;MA12FJ,uC;QAWkB,Q;QAAA,OAAa,SA+1FX,YA/1FF,SA+1FN,QAAQ,CA/1FW,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAAc,sBAAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,MAAM,gCAAuB,mDAAvB,C;O;KafV,C;iFAkBA,yB;MAAA,+C;MAAA,iE;MAG2FI,0D;MAh2FJ,uC;QAWkB,Q;QAAA,OAAa,SAq1FX,YAr1FF,SAq1FN,QAAQ,CAr1FW,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAAc,sBAAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,MAAM,gCAAuB,mDAAvB,C;O;KafV,C;+FAkBA,yB;MAAA,0D;MAAA,qC;QAOI,OAAe,YAAR,iBAAQ,EAAAY,OnC9sBM,KmC8sBIB,C;O;KAPnB,C;+FAUA,yB;MAAA,0D;MAAA,qC;QAOI,OAAe,YAAR,iBAAQ,EAAAY,OnB7sBQ,KmB6sBpB,C;O;KAPnB,C;+FAUA,yB;MAAA,0D;MAAA,qC;QAOI,OAAe,YAAR,iBAAQ,EAAAY,OpC1wBQ,KoC0wBpB,C;O;KAPnB,C;+FAUA,yB;MAAA,0D;MAAA,qC;QAOI,OAAe,YAAR,iBAAQ,EAAAY,OiCzwBU,KkCywBtB,C;O;KAPnB,C;IAUA,kC;MAQI,OAAW,mBAAJ,GAAe,IAAf,GAAyB,sBAAK,iBAAO,CAAP,IAAL,C;K;IAGpC,kC;MAQI,OAAW,mBAAJ,GAAe,IAAf,GAAyB,sBAAK,iBAAO,CAAP,IAAL,C;K;IAGpC,kC;MAQI,OAAW,mBAAJ,GAAe,IAAf,GAAyB,sBAAK,iBAAO,CAAP,IAAL,C;K;IAGpC,kC;MAQI,OAAW,mBAAJ,GAAe,IAAf,GAAyB,sBAAK,iBAAO,CAAP,IAAL,C;K;6FAGpC,yB;MAAA,+C;MAkuFI,0D;MALuFJ,uC;QASKB,Q;QAAA,OAAa,SAytFX,YAZtFF,SAytFN,QAAQ,CAztFW,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAAc,sBAAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,OAAO,I;O;KAbX,C;6FAGBA,yB;MAAA,+C;MA0tFI,0D;MA1tFJ,uC;QASKB,Q;QAAA,OAAa,SAitFX,YAjtFF,SAitFN,QAAQ,CAjtFW,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAAc,sBAAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,OAAO,I;O;KAbX,C;6FAGBA,yB;MAAA,+C;MA0sFI,0D;MA1sFJ,uC;QASKB,Q;QAAA,OAAa,SAisFX,YAjsFF,SAisFN,QAAQ,CAjsFW,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,cAAc,sBAAK,KAAL,C;UACd,IAAI,UAAU,OAAV,CAAJ,C;YAAwB,OAAO,O;;QAEEnC,OAAO,I;O;KAbX,C;qFAGBA,yB;MAAA,mC;MAAA,gD;MAAA,4B;QASI,OAAO,kBAAO,cAAP,C;O;KATX,C;qFAYA,yB;MAAA,mC;MAAA,gD;MAAA,4B;QASI,OAAO,kBAAO,cAAP,C;O;KATX,C;qFAYA,yB;MAAA,mC;MAAA,gD;MAAA,4B;QASI,OAAO,kBAAO,cAAP,C;O;KATX,C;IAYA,sC;MAQI,IAAI,mBAAJ,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,sBAAI,MAAO,iBAAQ,cAAR,CAAX,C;K;IAGX,sC;MAQI,IAAI,mBAAJ,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,sBAAI,MAAO,iBAAQ,cAAR,CAAX,C;K;IAGX,sC;MAQI,IAAI,mBAAJ,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,sBAAI,MAAO,iBAAQ,cAAR,CAAX,C;K;IAGX,sC;MAQI,IAAI,mBAAJ,C;QACI,MAAM,2BAAuB,iBAAvB,C;MACV,OAAO,sBAAI,MAAO,iBAAQ,cAAR,CAAX,C;K;iGAGX,yB;MAAA,mC;MAAA,4D;MAAA,4B;QAQI,OAAO,wBAAa,cAAb,C;O;KARX,C;iGAWA,yB;MAAA,mC;MAAA,4D;MAAA,4B;QAQI,OAAO,wBAAa,cAAb,C;O;KARX,C;iGAWA,yB;MAAA,mC;MAAA,4D;MAAA,4B;QAQI,OAAO,wBAAa,cAAb,C;O;KARX,C;IAWA,4C;MAOI,IAAI,mBAAJ,C;QACI,OAAO,I;MACX,OAAO,sBAAI,MAAO,iBAAQ,cAAR,CAAX,C;K;IAGX,4C;MAOI,IAAI,mBAAJ,C;QACI,OAAO,I;MACX,OAAO,sBAAI,MAAO,iBAAQ,cAAR,CAAX,C;K;IAGX,4C;MAOI,IAAI,mBAAJ,C;QACI,OAAO,I;MACX,OAAO,sBAAI,MAAO,iBAAQ,cAAR,CAAX,C;K;qFAGX,yB;MAAA,gD;MnCh8BA,6B;MmCg8BA,4B;QAOI,OnC77BmC,cmC67BpB,

OAAR,iBAAQ,CnC77BoB,C;O;KmCs7BvC,C;qFAUA,yB;MAAA,gD;MnB37BA,+B;MmB27BA,4B;QAOI,OnB  
x7BsC,emBw7BvB,OAAR,iBAAQ,CnBx7BuB,C;O;Kmbi7B1C,C;qFAUA,yB;MAAA,gD;MpClgCA,+B;MoCkg  
CA,4B;QAOI,OpC//BsC,eoC+/BvB,OAAR,iBAAQ,CpC//BuB,C;O;KoCw/B1C,C;qFAUA,yB;MAAA,gD;MIC//B  
A,iC;MkC+/BA,4B;QAOI,OIC5/ByC,gBkC4/B1B,OAAR,iBAAQ,CIC5/B0B,C;O;KkCq/B7C,C;qFAUA,yB;MAA  
A,kF;MAAA,iE;MAAA,wB;MAAA,8B;MAAA,uC;QASoB,UAST,M;QAXP,aAAoB,I;QACpB,YAAY,K;QACI,2  
B;QAAhB,OAAGB,cAAhB,C;UAGB,yB;UACZ,IAAI,UAAU,OAAV,CAAJ,C;YACI,IAAI,KAAJ,C;cAAW,MA  
AM,8BAAYB,gDAAzB,C;YACjB,SAAS,O;YACT,QAAQ,I;;QAGhB,IAAI,CAAC,KAAL,C;UAA,Y,MAAM,gCA  
AuB,mDAAvB,C;QAEIB,OAAO,0D;O;KAIBX,C;qFAqBA,yB;MAAA,kF;MAAA,iE;MAAA,0B;MAAA,8B;MA  
AA,uC;QASoB,UAST,M;QAXP,aAAqB,I;QACrB,YAAY,K;QACI,2B;QAAhB,OAAGB,cAAhB,C;UAGB,yB;U  
ACZ,IAAI,UAAU,OAAV,CAAJ,C;YACI,IAAI,KAAJ,C;cAAW,MAAM,8BAAYB,gDAAzB,C;YACjB,SAAS,O;Y  
ACT,QAAQ,I;;QAGhB,IAAI,CAAC,KAAL,C;UAA,Y,MAAM,gCAAuB,mDAAvB,C;QAEIB,OAAO,2D;O;KAIB  
X,C;qFAqBA,yB;MAAA,kF;MAAA,iE;MAAA,0B;MAAA,8B;MAAA,uC;QASoB,UAST,M;QAXP,aAAqB,I;QA  
CrB,YAAY,K;QACI,2B;QAAhB,OAAGB,cAAhB,C;UAGB,yB;UACZ,IAAI,UAAU,OAAV,CAAJ,C;YACI,IAAI  
,KAAJ,C;cAAW,MAAM,8BAAYB,gDAAzB,C;YACjB,SAAS,O;YACT,QAAQ,I;;QAGhB,IAAI,CAAC,KAAL,C;  
UAA,Y,MAAM,gCAAuB,mDAAvB,C;QAEIB,OAAO,2D;O;KAIBX,C;qFAqBA,yB;MAAA,kF;MAAA,iE;MAAA  
,4B;MAAA,8B;MAAA,uC;QASoB,UAST,M;QAXP,aAAsB,I;QACtB,YAAY,K;QACI,2B;QAAhB,OAAGB,cAAh  
B,C;UAGB,yB;UACZ,IAAI,UAAU,OAAV,CAAJ,C;YACI,IAAI,KAAJ,C;cAAW,MAAM,8BAAYB,gDAAzB,C;  
YACjB,SAAS,O;YACT,QAAQ,I;;QAGhB,IAAI,CAAC,KAAL,C;UAA,Y,MAAM,gCAAuB,mDAAvB,C;QAEIB,  
OAAO,4D;O;KAIBX,C;IAqBA,oC;MAMI,OAAW,mBAAQ,CAAZ,GAAe,sBAAK,CAAL,CAAF,GAA4B,I;K;IA  
GvC,oC;MAMI,OAAW,mBAAQ,CAAZ,GAAe,sBAAK,CAAL,CAAF,GAA4B,I;K;IAGvC,oC;MAMI,OAAW,mB  
AAQ,CAAZ,GAAe,sBAAK,CAAL,CAAF,GAA4B,I;K;IAGvC,oC;MAMI,OAAW,mBAAQ,CAAZ,GAAe,sBAAK  
,CAAL,CAAF,GAA4B,I;K;iGAGvC,gC;MASoB,Q;MAFhB,aAAoB,I;MACpB,YAAY,K;MACI,2B;MAAhB,OA  
gB,cAAhB,C;QAAgB,yB;QACZ,IAAI,UAAU,OAAV,CAAJ,C;UACI,IAAI,KAAJ,C;YAAW,OAAO,I;UACIB,SA  
AS,O;UACT,QAAQ,I;;MAGhB,IAAI,CAAC,KAAL,C;QAAY,OAAO,I;MACnB,OAAO,M;K;iGAGX,gC;MASoB  
,Q;MAFhB,aAAqB,I;MACrB,YAAY,K;MACI,2B;MAAhB,OAAGB,cAAhB,C;QAAgB,yB;QACZ,IAAI,UAAU,O  
AAV,CAAJ,C;UACI,IAAI,KAAJ,C;YAAW,OAAO,I;UACIB,SAAS,O;UACT,QAAQ,I;;MAGhB,IAAI,CAAC,KA  
AL,C;QAAY,OAAO,I;MACnB,OAAO,M;K;iGAGX,gC;MASoB,Q;MAFhB,aAAqB,I;MACrB,YAAY,K;MACI,2  
B;MAAhB,OAAGB,cAAhB,C;QAAgB,yB;QACZ,IAAI,UAAU,OAAV,CAAJ,C;UACI,IAAI,KAAJ,C;YAAW,OA  
AO,I;UACIB,SAAS,O;UACT,QAAQ,I;;MAGhB,IAAI,CAAC,KAAL,C;QAAY,OAAO,I;MACnB,OAAO,M;K;iG  
AGX,gC;MASoB,Q;MAFhB,aAAsB,I;MACtB,YAAY,K;MACI,2B;MAAhB,OAAGB,cAAhB,C;QAAgB,yB;QAC  
Z,IAAI,UAAU,OAAV,CAAJ,C;UACI,IAAI,KAAJ,C;YAAW,OAAO,I;UACIB,SAAS,O;UACT,QAAQ,I;;MAGhB  
,IAAI,CAAC,KAAL,C;QAAY,OAAO,I;MACnB,OAAO,M;K;IAGX,+B;MxBrhDI,IAAI,EwB+hDI,KAAK,CxB/h  
DT,CAAJ,C;QACI,cwB8hDc,sD;QxB7hDd,MAAM,gCAAYB,OAAQ,WAAjC,C;OwB8hDV,OAAO,uBAAoB,gB  
AAV,iBAAO,CAAP,IAAU,EAAC,CAAd,CAApB,C;K;IAGX,+B;MxBniDI,IAAI,EwB6iDI,KAAK,CxB7iDT,CA  
AJ,C;QACI,cwB4iDc,sD;QxB3iDd,MAAM,gCAAYB,OAAQ,WAAjC,C;OwB4iDV,OAAO,uBAAoB,gBAAV,iBAA  
AO,CAAP,IAAU,EAAC,CAAd,CAApB,C;K;IAGX,+B;MxBjjDI,IAAI,EwB2jDI,KAAK,CxB3jDT,CAAJ,C;QACI,  
cwB0jDc,sD;QxBzjDd,MAAM,gCAAYB,OAAQ,WAAjC,C;OwB0jDV,OAAO,uBAAoB,gBAAV,iBAAO,CAAP,IA  
AAU,EAAC,CAAd,CAApB,C;K;IAGX,+B;MxBjDI,IAAI,EwBykDI,KAAK,CxBzkDT,CAAJ,C;QACI,cwBwkDc,  
sD;QxBvkDd,MAAM,gCAAYB,OAAQ,WAAjC,C;OwBwkDV,OAAO,uBAAoB,gBAAV,iBAAO,CAAP,IAAU,EA  
AAC,CAAd,CAApB,C;K;IAGX,mC;MxB7kDI,IAAI,EwBulDI,KAAK,CxBvIDT,CAAJ,C;QACI,cwBslDc,sD;QxB  
rlDd,MAAM,gCAAYB,OAAQ,WAAjC,C;OwBslDV,OAAO,mBAAgB,gBAAV,iBAAO,CAAP,IAAU,EAAC,CAA  
d,CAAhB,C;K;IAGX,mC;MxB3lDI,IAAI,EwBqmDI,KAAK,CxBrmDT,CAAJ,C;QACI,cwBomDc,sD;QxBnmDd,  
MAAM,gCAAYB,OAAQ,WAAjC,C;OwBomDV,OAAO,mBAAgB,gBAAV,iBAAO,CAAP,IAAU,EAAC,CAAd,C  
AAhB,C;K;IAGX,mC;MxBzmDI,IAAI,EwBmnDI,KAAK,CxBnnDT,CAAJ,C;QACI,cwBknDc,sD;QxBjnDd,MA  
AM,gCAAYB,OAAQ,WAAjC,C;OwBknDV,OAAO,mBAAgB,gBAAV,iBAAO,CAAP,IAAU,EAAC,CAAd,CAAh  
B,C;K;IAGX,mC;MxBvnDI,IAAI,EwBioDI,KAAK,CxBjoDT,CAAJ,C;QACI,cwBgoDc,sD;QxB/nDd,MAAM,gC  
AAyB,OAAQ,WAAjC,C;OwBgoDV,OAAO,mBAAgB,gBAAV,iBAAO,CAAP,IAAU,EAAC,CAAd,CAAhB,C;K;  
mGAGX,yB;MAAA,4C;MAAA,qD;MAkqEI,8D;MAIqEJ,uC;QASl,iBAypEgB,cAAR,iBAAQ,CAzpEhB,WAA+B

,CAA/B,U;UACI,IAAI,CAAC,UAAU,sBAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL,C;;QAGf,OAAO,W;O;KAdX,C;mGAIbA,yB;MAAA,4C;MAAA,qD;MAypEI,8D;MAzpEJ,uC;QASI,iBAgpEgB,cAAR,iBAAQ,CAhpEhB,WAA+B,CAA/B,U;UACI,IAAI,CAAC,UAAU,sBAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL,C;;QAGf,OAAO,W;O;KAdX,C;mGAIbA,yB;MAAA,4C;MAAA,qD;MAgpEI,8D;MAhpEJ,uC;QASI,iBAuoEgB,cAAR,iBAAQ,CAvoEhB,WAA+B,CAA/B,U;UACI,IAAI,CAAC,UAAU,sBAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL,C;;QAGf,OAAO,W;O;KAdX,C;mGAIbA,yB;MAAA,4C;MAAA,qD;MAuoEI,8D;MAvoEJ,uC;QASI,iBA8nEgB,cAAR,iBAAQ,CA9nEhB,WAA+B,CAA/B,U;UACI,IAAI,CAAC,UAAU,sBAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL,C;;QAGf,OAAO,W;O;KAdX,C;2FAiBA,yB;MAAA,+D;MAAA,uC;QAWiB,Q;QAFb,eAAe,K;QACf,WAAW,gB;QACE,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,IAAI,QAAJ,C;YACI,IAAK,WAAI,IAAJ,C;eACJ,IAAI,CAAC,UAAU,IAAV,CAAL,C;YACD,IAAK,WAAI,IAAJ,C;YACL,WAAW,I;;QAEhB,OAAO,I;O;KAIBX,C;2FAqBA,yB;MAAA,+D;MAAA,uC;QAWiB,Q;QAFb,eAAe,K;QACf,WAAW,gB;QACE,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,IAAI,QAAJ,C;YACI,IAAK,WAAI,IAAJ,C;eACJ,IAAI,CAAC,UAAU,IAAV,CAAL,C;YACD,IAAK,WAAI,IAAJ,C;YACL,WAAW,I;;QAEhB,OAAO,I;O;KAIBX,C;2FAqBA,yB;MAAA,+D;MAAA,uC;QAWiB,Q;QAFb,eAAe,K;QACf,WAAW,gB;QACE,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,IAAI,QAAJ,C;YACI,IAAK,WAAI,IAAJ,C;eACJ,IAAI,CAAC,UAAU,IAAV,CAAL,C;YACD,IAAK,WAAI,IAAJ,C;YACL,WAAW,I;;QAEhB,OAAO,I;O;KAIBX,C;2FAqBA,yB;MAAA,+D;MAAA,uC;QAWiB,Q;QAFb,eAAe,K;QACf,WAAW,gB;QACE,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,IAAI,QAAJ,C;YACI,IAAK,WAAI,IAAJ,C;eACJ,IAAI,CAAC,UAAU,IAAV,CAAL,C;YACD,IAAK,WAAI,IAAJ,C;YACL,WAAW,I;;QAEhB,OAAO,I;O;KAIBX,C;qFAqBA,yB;MAAA,+D;MAAA,uC;QASW,kBAAS,gB;QAgRA,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UAAM,IAhRa,SAgRT,CAAU,OAAV,CAAJ,C;YAAwB,WAAY,WAAI,OAAJ,C;;QAhR1D,OAIRO,W;O;KA1RX,C;qFAYA,yB;MAAA,+D;MAAA,uC;QASW,kBAAS,gB;QAIrA,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UAAM,IAjRc,SAiRV,CAAU,OAAV,CAAJ,C;YAAwB,WAAY,WAAI,OAAJ,C;;QAjR1D,OAKRO,W;O;KA3RX,C;qFAYA,yB;MAAA,+D;MAAA,uC;QASW,kBAAS,gB;QAKrA,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UAAM,IAIRc,SAkRV,CAAU,OAAV,CAAJ,C;YAAwB,WAAY,WAAI,OAAJ,C;;QAIR1D,OAmRO,W;O;KA5RX,C;qFAYA,yB;MAAA,+D;MAAA,uC;QASW,kBAAS,gB;QAmRA,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UAAM,IAAnRe,SAmRX,CAAU,OAAV,CAAJ,C;YAAwB,WAAY,WAAI,OAAJ,C;;QAnR1D,OAoRO,W;O;KA7RX,C;kGAYA,yB;MAAA,+D;MAAA,uC;QAWW,kBAAGB,gB;QAm5HV,gB;QADb,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UA11HT,IAzDsC,SAyDIC,EA01HkB,cA11HIB,EA01HkB,sBA11HIB,WA01H2B,IA11H3B,CAAJ,C;YAA2C,sBA01HZ,IA11HY,C;;QAZD/C,OA2DO,W;O;KATeX,C;mGAcA,yB;MAAA,+D;MAAA,uC;QAWW,kBAAGB,gB;QAK5HV,gB;QADb,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UA11HT,IA5DuC,SA4DnC,EAs1HkB,cAt1HIB,EAs1HkB,sBA11HIB,WAs1H2B,IA11H3B,CAAJ,C;YAA2C,sBA11HZ,IA11HY,C;;QA5D/C,OA8DO,W;O;KAZeX,C;mGAcA,yB;MAAA,+D;MAAA,uC;QAWW,kBAAGB,gB;QAI5HV,gB;QADb,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UA11HT,IA/DuC,SA+DnC,EAK1HkB,cA11HIB,EAK1HkB,sBA11HIB,WAK1H2B,IA11H3B,CAAJ,C;YAA2C,sBAK1HZ,IA11HY,C;;QA/D/C,OAIeO,W;O;KA5EX,C;mGAcA,yB;MAAA,+D;MAAA,uC;QAWW,kBAAGB,gB;QAG5HV,gB;QADb,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UA90HT,IAIEwC,SAkEpC,EA80HkB,cA90HIB,EA80HkB,sBA90HIB,WA80H2B,IA90H3B,CAAJ,C;YAA2C,sBA80HZ,IA90HY,C;;QAIe/C,OAoEO,W;O;KA/EX,C;uGAcA,6C;MA52HiB,gB;MADb,YAAY,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QA11HT,IAAI,WA01HkB,cA11HIB,EA01HkB,sBA11HIB,WA01H2B,IA11H3B,CAAJ,C;UAA2C,sBA01HZ,IA11HY,C;;MAE/C,OAAO,W;K;uGAGX,6C;MAK2HiB,gB;MADb,YAAY,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QA11HT,IAAI,WAK1HkB,cA11HIB,EAK1HkB,sBA11HIB,WAK1H2B,IA11H3B,CAAJ,C;UAA2C,sBAK1HZ,IA11HY,C;;MAE/C,OAAO,W;K;uGAGX,6C;MA01HiB,gB;MADb,YAAY,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QA90HT,IAAI,WA80HkB,cA90HIB,EA80HkB,sBA90HIB,WA80H2B,IA90H3B,CAAJ,C;UAA2C,sBA80HZ,IA90HY,C;;MAE/C,OAAO,W;K;2FAGX,yB;MAAA,+D;MAAA,uC;QASW,kBAAY,gB;QAGDH,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UAAM,IAAI,CAhDY,SAgDX,CAAU,OAAV,CAAL,C;YAAyB,WAAY,WAAI,OAAJ,C;;QAhD3D,OAI DO,W;O;KA1DX,C;2FAYA,yB;MAAA,

+D;MAAA,uC;QASW,kBAAY,gB;QAI DH,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UAAM,IAAI,C  
AjDa,SAiDZ,CAAU,OAAV,CAAL,C;YAAyB,WAAy,WAAI,OAAJ,C;;QAjD3D,OAKDO,W;O;KA3DX,C;2FAY  
A,yB;MAAA,+D;MAAA,uC;QASW,kBAAY,gB;QAKDH,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;U  
AAM,IAAI,CAIDa,SAkDZ,CAAU,OAAV,CAAL,C;YAAyB,WAAy,WAAI,OAAJ,C;;QAID3D,OAmDO,W;O;K  
A5DX,C;2FAYA,yB;MAAA,+D;MAAA,uC;QASW,kBAAY,gB;QAmDH,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,  
C;UAAgB,yB;UAAM,IAAI,CAnDc,SAmDb,CAAU,OAAV,CAAL,C;YAAyB,WAAy,WAAI,OAAJ,C;;QAnD3D,  
OAO DO,W;O;KA7DX,C;+FAYA,6C;MASoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI  
,CAAC,UAAU,OAAV,CAAL,C;UAAyB,WAAy,WAAI,OAAJ,C;;MAC3D,OAAO,W;K;+FAGX,6C;MASoB,Q;  
MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,CAAC,UAAU,OAAV,CAAL,C;UAAyB,WAA  
Y,WAAI,OAAJ,C;;MAC3D,OAAO,W;K;+FAGX,6C;MASoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,  
yB;QAAM,IAAI,CAAC,UAAU,OAAV,CAAL,C;UAAyB,WAAy,WAAI,OAAJ,C;;MAC3D,OAAO,W;K;+FAGX  
,6C;MASoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,CAAC,UAAU,OAAV,CAAL,C;  
UAAyB,WAAy,WAAI,OAAJ,C;;MAC3D,OAAO,W;K;yFAGX,6C;MASoB,Q;MAAA,2B;MAAhB,OAAgB,cAA  
hB,C;QAAgB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,WAAy,WAAI,OAAJ,C;;MAC1D,OAAO,W;K;  
yFAGX,6C;MASoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;  
UAAwB,WAAy,WAAI,OAAJ,C;;MAC1D,OAAO,W;K;yFAGX,6C;MASoB,Q;MAAA,2B;MAAhB,OAAgB,cAA  
hB,C;QAAgB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,WAAy,WAAI,OAAJ,C;;MAC1D,OAAO,W;K;  
yFAGX,6C;MASoB,Q;MAAA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;  
UAAwB,WAAy,WAAI,OAAJ,C;;MAC1D,OAAO,W;K;IAGX,sC;MAMI,IAAI,OAAQ,UAAZ,C;QAAuB,OhCvjE  
e,W;OgCwjEtC,OAA4D,SA0iDrD,cAAkB,cAAR,iBAAQ,EA1iDN,OAAQ,MA0iDF,EA1iDS,OAAQ,aAAR,GAA  
uB,CAA vB,IA0iDT,CAA1B,CA1iDqD,C;K;IAGhE,sC;MAMI,IAAI,OAAQ,UAAZ,C;QAAuB,OhCjkEe,W;OgCkk  
EtC,OAA4D,SAgjDrD,eAAmB,cAAR,iBAAQ,EAhjDP,OAAQ,MAgjDD,EAhjDQ,OAAQ,aAAR,GAAuB,CAA vB  
,IAgjDR,CAAnB,CAhjDqD,C;K;IAGhE,sC;MAMI,IAAI,OAAQ,UAAZ,C;QAAuB,OhC3kEe,W;OgC4kEtC,OAA  
4D,UAsjDrD,eAAmB,cAAR,iBAAQ,EA tjDP,OAAQ,MA sjDD,EA tjDQ,OAAQ,aAAR,GAAuB,CAA vB,IASjDR,C  
AAnB,CAtjDqD,C;K;IAGhE,sC;MAMI,IAAI,OAAQ,UAAZ,C;QAAuB,OhCrlEe,W;OgCslEtC,OAA4D,UA4jDrD  
,gBAAoB,cAAR,iBAAQ,EA5jDR,OAAQ,MA4jDA,EA5jDO,OAAQ,aAAR,GAAuB,CAA vB,IA4jDP,CAApB,CA  
5jDqD,C;K;IAGhE,sC;MAskB,Q;MAHd,WAAmB,wBAAR,OAAQ,EA AwB,EA AxB,C;MACnB,IAAI,SAAQ,CA  
AZ,C;QAAe,OAAO,W;MACtB,WAAW,iBAAGB,IAAhB,C;MACG,yB;MAAd,OAAc,cAA d,C;QAAc,uB;QACV,I  
AAK,WAAI,sBAAL,KA AJ,CAAJ,C;;MAET,OAAO,I;K;IAGX,sC;MAskB,Q;MAHd,WAAmB,wBAAR,OAAQ,EA  
AwB,EA AxB,C;MACnB,IAAI,SAAQ,CAAZ,C;QAAe,OAAO,W;MACtB,WAAW,iBAAiB,IAAjB,C;MACG,yB  
;MAAd,OAAc,cAA d,C;QAAc,uB;QACV,IAAK,WAAI,sBAAL,KA AJ,CAAJ,C;;MAET,OAAO,I;K;IAGX,sC;MA  
SkB,Q;MAHd,WAAmB,wBAAR,OAAQ,EA AwB,EA AxB,C;MACnB,IAAI,SAAQ,CAAZ,C;QAAe,OAAO,W;M  
ACtB,WAAW,iBAAiB,IAAjB,C;MACG,yB;MAAd,OAAc,cAA d,C;QAAc,uB;QACV,IAAK,WAAI,sBAAL,KA AJ  
,CAAJ,C;;MAET,OAAO,I;K;IAGX,sC;MAskB,Q;MAHd,WAAmB,wBAAR,OAAQ,EA AwB,EA AxB,C;MACnB,  
IAAI,SAAQ,CAAZ,C;QAAe,OAAO,W;MACtB,WAAW,iBAAkB,IAAIB,C;MACG,yB;MAAd,OAAc,cAA d,C;Q  
AAc,uB;QACV,IAAK,WAAI,sBAAL,KA AJ,CAAJ,C;;MAET,OAAO,I;K;IAGX,2C;MAMI,OAAO,cAAkB,aAAR,  
iBAAQ,EA AW,OAAX,CAAIB,C;K;IAGX,2C;MAMI,OAAO,eAAmB,aAAR,iBAAQ,EA AW,OAAX,CAAnB,C;K  
;IAGX,2C;MAMI,OAAO,eAAmB,aAAR,iBAAQ,EA AW,OAAX,CAAnB,C;K;IAGX,2C;MAMI,OAAO,gBAAoB,  
aAAR,iBAAQ,EA AW,OAAX,CAApB,C;K;IAGX,2C;MAMI,OAAO,cAAkB,cAAR,iBAAQ,EA AW,OAAX,CAAI  
B,C;K;IAGX,2C;MAMI,OAAO,eAAmB,cAAR,iBAAQ,EA AW,OAAX,CAAnB,C;K;IAGX,2C;MAMI,OAAO,eA  
AmB,aAAR,iBAAQ,EA AW,OAAX,CAAnB,C;K;IAGX,2C;MAMI,OAAO,gBAAoB,cAAR,iBAAQ,EA AW,OA  
AX,CAApB,C;K;IAGX,+B;MAGBiB,Q;MxB7xEb,IAAI,EwBuxEI,KA AK,CxBvxET,CAAJ,C;QACI,cwBsxEc,sD;Q  
xBrxE d,MAAM,gCAAYB,OAAQ,WAAjC,C;OwBsxEV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,IAAI,  
KA AK,cAAT,C;QAAe,OAAO,mB;MACtB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,sBAAK,CAAL,CAAP,  
C;MACnB,YAAY,C;MACZ,WAAW,iBAAGB,CAAhB,C;MACE,2B;MAAb,OAAa,cAA b,C;QAAa,sB;QACT,IAA  
K,WAAI,IAAJ,C;QACL,IAAI,mCAAW,CAAF,C;UACI,K;;MAER,OAAO,I;K;IAGX,+B;MAGBiB,Q;MxB7yEd,IA  
AI,EwB+yEI,KA AK,CxB/yET,CAAJ,C;QACI,cwB8yEc,sD;QxB7yEd,MAAM,gCAAYB,OAAQ,WAAjC,C;OwB8  
yEV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,IAAI,KA AK,cAAT,C;QAAe,OAAO,mB;MACtB,IAAI,M

AAK,CAAT,C;QAAY,OAAO,OAAO,sBAAK,CAAL,CAAP,C;MACnB,YAAY,C;MACZ,WAAW,iBAAiB,CAAjB,C;MACE,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,IAAK,WAAI,IAAJ,C;QACL,IAAI,mCAAW,CAAf,C;UACI,K;;MAER,OAAO,I;K;IAGX,+B;MAGBiB,Q;MxB70Eb,IAAI,EwBu0EI,KAAK,CxBv0ET,CAAJ,C;QACI,cwBs0Ec,sD;QxB70Ed,MAAM,gCAAYB,OAAQ,WAAjC,C;OwBs0EV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,IAAI,KAAK,cAAT,C;QAAe,OAAO,mB;MACtB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,sBAAK,CAAL,CAAP,C;MACnB,YAAY,C;MACZ,WAAW,iBAAiB,CAAjB,C;MACE,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,IAAK,WAAI,IAAJ,C;QACL,IAAI,mCAAW,CAAf,C;UACI,K;;MAER,OAAO,I;K;IAGX,+B;MAGBiB,Q;MxB72Eb,IAAI,EwB+1EI,KAAK,CxB/1ET,CAAJ,C;QACI,cwB81Ec,sD;QxB71Ed,MAAM,gCAAYB,OAAQ,WAAjC,C;OwB81EV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,IAAI,KAAK,cAAT,C;QAAe,OAAO,mB;MACtB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,sBAAK,CAAL,CAAP,C;MACnB,YAAY,C;MACZ,WAAW,iBAAkB,CAAIB,C;MACE,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,IAAK,WAAI,IAAJ,C;QACL,IAAI,mCAAW,CAAf,C;UACI,K;;MAER,OAAO,I;K;IAGX,mC;MxB72EI,IAAI,EwBu3EI,KAAK,CxBv3ET,CAAJ,C;QACI,cwBs3Ec,sD;QxB73Ed,MAAM,gCAAYB,OAAQ,WAAjC,C;OwBs3EV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,WAAW,c;MACX,IAAI,KAAK,IAAT,C;QAAe,OAAO,mB;MACtB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,sBAAK,OAAO,CAAP,IAAL,CAAP,C;MACnB,WAAW,iBAAgB,CAAhB,C;MACX,iBAAc,OAAO,CAAP,IAAd,UAA6B,IAA7B,U;QACI,IAAK,WAAI,sBAAK,KAAL,CAAJ,C;MACT,OAAO,I;K;IAGX,mC;MxB14EI,IAAI,EwB44EI,KAAK,CxB54ET,CAAJ,C;QACI,cwB24Ec,sD;QxB14Ed,MAAM,gCAAYB,OAAQ,WAAjC,C;OwB24EV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,WAAW,c;MACX,IAAI,KAAK,IAAT,C;QAAe,OAAO,mB;MACtB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,sBAAK,OAAO,CAAP,IAAL,CAAP,C;MACnB,WAAW,iBAAiB,CAAjB,C;MACX,iBAAc,OAAO,CAAP,IAAd,UAA6B,IAA7B,U;QACI,IAAK,WAAI,sBAAK,KAAL,CAAJ,C;MACT,OAAO,I;K;IAGX,mC;MxBv5EI,IAAI,EwBi6EI,KAAK,CxBj6ET,CAAJ,C;QACI,cwBg6Ec,sD;QxB/5Ed,MAAM,gCAAYB,OAAQ,WAAjC,C;OwBg6EV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,WAAW,c;MACX,IAAI,KAAK,IAAT,C;QAAe,OAAO,mB;MACtB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,sBAAK,OAAO,CAAP,IAAL,CAAP,C;MACnB,WAAW,iBAAiB,CAAjB,C;MACX,iBAAc,OAAO,CAAP,IAAd,UAA6B,IAA7B,U;QACI,IAAK,WAAI,sBAAK,KAAL,CAAJ,C;MACT,OAAO,I;K;IAGX,mC;MxB56EI,IAAI,EwBs7EI,KAAK,CxBt7ET,CAAJ,C;QACI,cwBq7Ec,sD;QxBp7Ed,MAAM,gCAAYB,OAAQ,WAAjC,C;OwBq7EV,IAAI,MAAK,CAAT,C;QAAY,OAAO,W;MACnB,WAAW,c;MACX,IAAI,KAAK,IAAT,C;QAAe,OAAO,mB;MACtB,IAAI,MAAK,CAAT,C;QAAY,OAAO,OAAO,sBAAK,OAAO,CAAP,IAAL,CAAP,C;MACnB,WAAW,iBAAkB,CAAIB,C;MACX,iBAAc,OAAO,CAAP,IAAd,UAA6B,IAA7B,U;QACI,IAAK,WAAI,sBAAK,KAAL,CAAJ,C;MACT,OAAO,I;K;mGAGX,yB;MAAA,4C;MAAA,gD;MA52CI,8D;MAf2CJ,uC;QASI,iBA61CgB,cAAR,iBAAQ,CA71ChB,WAA+B,CAA/B,U;UACI,IAAI,CAAC,UAAU,sBAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL,C;;QAGf,OAAO,iB;O;KAdX,C;mGaiBA,yB;MAAA,4C;MAAA,gD;MA61CI,8D;MA71CJ,uC;QASI,iBAo1CgB,cAAR,iBAAQ,Cap1ChB,WAA+B,CAA/B,U;UACI,IAAI,CAAC,UAAU,sBAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL,C;;QAGf,OAAO,iB;O;KAdX,C;mGaiBA,yB;MAAA,4C;MAAA,gD;MAo1CI,8D;MAp1CJ,uC;QASI,iBA20CgB,cAAR,iBAAQ,CA30ChB,WAA+B,CAA/B,U;UACI,IAAI,CAAC,UAAU,sBAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL,C;;QAGf,OAAO,iB;O;KAdX,C;mGaiBA,yB;MAAA,4C;MAAA,gD;MA20CI,8D;MA30CJ,uC;QASI,iBAk0CgB,cAAR,iBAAQ,CA10ChB,WAA+B,CAA/B,U;UACI,IAAI,CAAC,UAAU,sBAAK,KAAL,CAAV,CAAL,C;YACI,OAAO,gBAAK,QAAQ,CAAR,IAAL,C;;QAGf,OAAO,iB;O;KAdX,C;2FAiBA,yB;MAAA,+D;MAAA,uC;QAUiB,Q;QADb,WAAW,gB;QACE,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,IAAI,CAAC,UAAU,IAAV,CAAL,C;YACI,K;UACJ,IAAK,WAAI,IAAJ,C;;QAET,OAAO,I;O;KafX,C;2FAkBA,yB;MAAA,+D;MAAA,uC;QAUiB,Q;QADb,WAAW,gB;QACE,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,IAAI,CAAC,UAAU,IAAV,CAAL,C;YACI,K;UACJ,IAAK,WAAI,IAAJ,C;;QAET,OAAO,I;O;KafX,C;2FAkBA,yB;MAAA,+D;MAAA,uC;QAUiB,Q;QADb,WAAW,gB;QACE,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,IAAI,CAAC,UAAU,IAAV,CAAL,C;YACI,K;UACJ,IAAK,WAAI,IAAJ,C;;QAET,OAAO,I;O;KafX,C;2FAkBA,yB;MAAA,+D;MAAA,uC;QAUiB,Q;QADb,WAAW,gB;QACE,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,IAAI,CAAC,UAAU,IAAV,CAAL,C;YACI,K;UACJ,IAAK,WAAI,IAAJ,C;;QAET,OAAO,I;O;KafX,C;uFAkBA,yB;MAAA,kD;MAAA,4B;QAOY,QAAR,iBAAQ,C;O;KAPZ,C;uFAUA,yB;MAAA,kD;MAAA,4B;QAOY,QAAR,iBAAQ,C;O;KAPZ,C;uFAUA,yB;MAAA,kD;MAAA,4B;QAOY,QAAR,iBAAQ,C;O;

KAPZ,C;uFAUA,yB;MAAA,kD;MAAA,4B;QAOY,QAAR,iBAAQ,C;O;KAPZ,C;uFAUA,yB;MAAA,kD;MAAA,gD;QAaY,QAAR,iBAAQ,EAAQ,SAAR,EAAmB,OAAAnB,C;O;KAbZ,C;uFAGBA,yB;MAAA,kD;MAAA,gD;QAaY,QAAR,iBAAQ,EAAQ,SAAR,EAAmB,OAAAnB,C;O;KAbZ,C;uFAGBA,yB;MAAA,kD;MAAA,gD;QAaY,QAAR,iBAAQ,EAAQ,SAAR,EAAmB,OAAAnB,C;O;KAbZ,C;IAgBA,gC;MAMI,IAAI,mBAAJ,C;QAaE,OAAO,W;MACtB,WAAW,0B;MACN,WAAL,IAAK,C;MACL,OAAO,I;K;IAGX,gC;MAMI,IAAI,mBAAJ,C;QAaE,OAAO,W;MACtB,WAAW,0B;MACN,WAAL,IAAK,C;MACL,OAAO,I;K;IAGX,gC;MAMI,IAAI,mBAAJ,C;QAaE,OAAO,W;MACtB,WAAW,0B;MACN,WAAL,IAAK,C;MACL,OAAO,I;K;IAGX,gC;MAMI,IAAI,mBAAJ,C;QAaE,OAAO,W;MACtB,WAAW,0B;MACN,WAAL,IAAK,C;MACL,OAAO,I;K;KAGGX,yB;MAAA,8D;MAAA,uC;MAAA,4B;QAOI,OAAO,mBAaB,cAAR,iBAAQ,CAAIb,C;O;KAPX,C;kGAUA,yB;MAAA,8D;MAAA,yC;MAAA,4B;QAOI,OAAO,mBAaB,cAAR,iBAAQ,CAAIb,C;O;KAPX,C;mGAUA,yB;MAAA,8D;MAAA,2C;MAAA,4B;QAOI,OAAO,qBAAoB,cAAR,iBAAQ,CAApB,C;O;KAPX,C;IAUA,+B;MAMI,sBAAQ,4BAAR,C;K;IAGJ,+B;MAMI,sBAAQ,4BAAR,C;K;IAGJ,+B;MAMI,sBAAQ,4BAAR,C;K;IAGJ,uC;MAQI,aA8+BgB,gBAAR,iBAAQ,CA9+BhB,OAA2B,CAA3B,M;QACI,QAAQ,MAAO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,WAAW,sBAAK,CAAL,C;QACX,sBAAK,CAAL,EAAU,sBAAK,CAAL,CAAV,C;QACA,sBAAK,CAAL,EAAU,IAAV,C;;K;IAIR,uC;MAQI,aAs+BgB,gBAAR,iBAAQ,CAt+BhB,OAA2B,CAA3B,M;QACI,QAAQ,MAAO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,WAAW,sBAAK,CAAL,C;QACX,sBAAK,CAAL,EAAU,sBAAK,CAAL,CAAV,C;QACA,sBAAK,CAAL,EAAU,IAAV,C;;K;IAIR,uC;MAQI,aA89BgB,gBAAR,iBAAQ,CA99BhB,OAA2B,CAA3B,M;QACI,QAAQ,MAAO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,WAAW,sBAAK,CAAL,C;QACX,sBAAK,CAAL,EAAU,sBAAK,CAAL,CAAV,C;QACA,sBAAK,CAAL,EAAU,IAAV,C;;K;IAIR,uC;MAQI,aAs9BgB,gBAAR,iBAAQ,CAt9BhB,OAA2B,CAA3B,M;QACI,QAAQ,MAAO,iBAAQ,IAAI,CAAJ,IAAR,C;QACf,WAAW,sBAAK,CAAL,C;QACX,sBAAK,CAAL,EAAU,sBAAK,CAAL,CAAV,C;QACA,sBAAK,CAAL,EAAU,IAAV,C;;K;IAIR,sC;MAMI,IAAI,iBAAO,CAAX,C;QACI,iB;QApSI,UAAR,iBAAQ,C;Q;IAySZ,sC;MAMI,IAAI,iBAAO,CAAX,C;QACI,iB;QAtSI,UAAR,iBAAQ,C;Q;IA2SZ,sC;MAMI,IAAI,iBAAO,CAAX,C;QACI,iB;QAxSI,UAAR,iBAAQ,C;Q;IA6SZ,sC;MAMI,IAAI,iBAAO,CAAX,C;QACI,iB;QA1SI,UAAR,iBAAQ,C;Q;IA+SZ,6B;MAMoB,kBA+nBT,cAAU,iBvB58EO,QuB48EjB,C;MA/nBiB,mB;MAAxB,OAAiC,SrBv3F1B,WqBu3F0B,C;K;IAGrC,8B;MAMoB,kBAkoBT,eAAmB,UAAR,iBAAQ,CAAnB,C;MAl0BiB,mB;MAAxB,OAAiC,SrBh4F1B,WqBg4F0B,C;K;IAGrC,8B;MAMoB,kBAqoBT,eAAW,iBvBx/EM,QuBw/EjB,C;MAroBiB,mB;MAAxB,OAAiC,Urbz4F1B,WqBy4F0B,C;K;IAGrC,8B;MAMoB,kBAwoBT,gBAAY,iBvB1/EK,QuB0/EjB,C;MAxoBiB,mB;MAAxB,OAAiC,Urb15F1B,WqBk5F0B,C;K;IAGrC,kC;MAMI,IAAI,mBAAJ,C;QAaE,OAAO,S;MACD,kBA0IbD,cA11BA,SA0IBU,QvB58EO,QuB48EjB,C;MA11BsB,mB;MAA7B,OrB55FO,W;K;IqB+5FX,kC;MAMI,IAAI,mBAAJ,C;QAaE,OAAO,S;MACD,kBA4IbD,eAAmB,UA5IbN,SA4IBW,QAAQ,CAAnB,C;MA5IBsB,mB;MAA7B,OrBt6FO,W;K;IqBy6FX,kC;MAMI,IAAI,mBAAJ,C;QAaE,OAAO,S;MACD,kBA8IbD,eA9IbA,SA8IBW,QvBx/EM,QuBw/EjB,C;MA9IBsB,mB;MAA7B,OrBh7FO,W;K;IqBm7FX,mC;MAMI,IAAI,mBAAJ,C;QAaE,OAAO,S;MACD,kBAgmBd,gBAhmBA,SAGmBY,QvB1/EK,QuB0/EjB,C;MAhmBsB,mB;MAA7B,OrB17FO,W;K;IqB67FX,4C;MAMI,IAAI,mBAAJ,C;QAaE,OAAO,S;MACD,kBAkjBd,cAljBA,SAkjBU,QvB58EO,QuB48EjB,C;MAIjBsB,8B;MAA7B,OrBp8FO,W;K;IqBu8FX,4C;MAMI,IAAI,mBAAJ,C;QAaE,OAAO,S;MACD,kBAojBd,eAAmB,UApjBnB,SAojBW,QAAQ,CAAnB,C;MApjBsB,8B;MAA7B,OrB98FO,W;K;IqBi9FX,4C;MAMI,IAAI,mBAAJ,C;QAaE,OAAO,S;MACD,kBAsjBd,eAtjBA,SAsjBW,QvBx/EM,QuBw/EjB,C;MATjBsB,8B;MAA7B,OrBx9FO,W;K;IqB29FX,6C;MAMI,IAAI,mBAAJ,C;QAaE,OAAO,S;MACD,kBAwjBd,gBAxjBA,SAwjBY,QvB1/EK,QuB0/EjB,C;MAxjBsB,8B;MAA7B,OrBl+FO,W;K;IqBq+FX,uC;MAQoB,kBAygBT,cAAU,iBvB58EO,QuB48EjB,C;MAZgBiB,mB;MAAxB,OAAiC,YrB7+F1B,WqB6+F0B,C;K;IAGrC,wC;MAQoB,kBA0gBT,eAAmB,UAAR,iBAAQ,CAAnB,C;MA1gBiB,mB;MAAxB,OAAiC,YrBx/F1B,WqBw/F0B,C;K;IAGrC,wC;MAQoB,kBA2gBT,eAAW,iBvBx/EM,QuBw/EjB,C;MA3gBiB,mB;MAAxB,OAAiC,YrBngG1B,WqBmgG0B,C;K;IAGrC,wC;MAQoB,kBA4gBT,gBAAY,iBvB1/EK,QuB0/EjB,C;MA5gBiB,mB;MAAxB,OAAiC,YrB9gG1B,WqB8gG0B,C;K;4FAGrC,qB;MAQI,OAAO,iB;K;0FAGX,qB;MAQI,OAAO,iB;K;4FA+BX,qB;MAQI,OAAO,iB;K;8FAGX,qB;MAQI,OAAO,iB;K;8FAGX,yB;MAAA,yC;MAAA,4B;QAQI,OAAO,oBAAW,SAAX,C;O;KARX,C;4FAWA,yB;MAAA,uC;MAAA,4B;QAQI,OAAO,mBAAU,S

AAV,C;O;KARX,C;8FAWA,yB;MAAA,yC;MAAA,4B;QAQI,OAAO,oBAAW,SAAX,C;O;KARX,C;gGAWA,yB ;MAAA,2C;MAAA,4B;QAQI,OAAO,qBAAY,SAAZ,C;O;KARX,C;IAWA,2C;MASI,OAAy,gBAAL,SAAK,EA Ac,KAAAd,C;K;IAGhB,2C;MASI,OAAy,gBAAL,SAAK,EAAC,KAAAd,C;K;IAGhB,2C;MASI,OAAy,gBAAL,SA AK,EAAC,KAAAd,C;K;IAGhB,2C;MASI,OAAy,gBAAL,SAAK,EAAC,KAAAd,C;K;IAGhB,2C;MAOI,OAAqB,cA Ad,4CAAc,EAAC,oCAAd,C;K;IAGzB,2C;MAOI,OAAqB,cAAd,4CAAc,EAAC,oCAAd,C;K;IAGzB,2C;MAOI,O AAqB,cAAd,4CAAc,EAAC,oCAAd,C;K;IAGzB,2C;MAOI,OAAqB,cAAd,4CAAc,EAAC,oCAAd,C;K;IAGzB,sC; MAQI,OAAy,kBAAL,SAAK,C;K;IAGhB,sC;MAQI,OAAy,kBAAL,SAAK,C;K;IAGhB,sC;MAQI,OAAy,kBAAL,SAAK,C;K;IAGhB,sC;MAQI,OAAy,kBAAL,SAAK,C;K;IAGhB,sC;MAMI,OAAqB,gBAAd,4CAAc,C;K;IAGz B,sC;MAMI,OAAqB,gBAAd,4CAAc,C;K;IAGzB,sC;MAMI,OAAqB,gBAAd,4CAAc,C;K;IAGzB,sC;MAMI,OA AqB,gBAAd,4CAAc,C;K;IAGzB,sC;MAUI,OAAy,kBAAL,SAAK,C;K;IAGhB,sC;MAUI,OAAy,kBAAL,SAAK, C;K;IAGhB,sC;MAUI,OAAy,kBAAL,SAAK,C;K;IAGhB,sC;MAUI,OAAy,kBAAL,SAAK,C;K;IAGhB,sC;MAQ W,Q;MAAP,OAAO,sDAAmB,IAAnB,EAAYB,GAAzB,EAA8B,GAA9B,2BAAsC,M;K;IAGjD,sC;MAQW,Q;MA AP,OAAO,sDAAmB,IAAnB,EAAYB,GAAzB,EAA8B,GAA9B,2BAAsC,M;K;IAGjD,sC;MAQW,Q;MAAP,OAA O,sDAAmB,IAAnB,EAAYB,GAAzB,EAA8B,GAA9B,2BAAsC,M;K;IAGjD,sC;MAQW,Q;MAAP,OAAO,sDAA mB,IAAnB,EAAYB,GAAzB,EAA8B,GAA9B,2BAAsC,M;K;sFAGjD,yB;MvBxhFA,8C;MuBwhFA,kF;QAmB6D, iC;UAAA,oBAAYB,C;QAAG,0B;UAAA,aAAkB,C;QAAG,wB;UAAA,WAAgB,c;QvBvhF1H,UuBwhFA,iBvBxh FA,EuBwhFiB,WAAy,QvBxhF7B,EuBwhFsC,iBvBxhFtC,EuBwhFyD,UvBxhFzD,EuBwhFqE,QvBxhFrE,C;QuB yhFA,OAAO,W;O;KArBX,C;wFAwBA,yB;MvBxhFA,8C;MuBwhFA,kF;QAmB+D,iC;UAAA,oBAAYB,C;QAA G,0B;UAAA,aAAkB,C;QAAG,wB;UAAA,WAAgB,c;QvBvhF5H,UuBwhFA,iBvBxhFA,EuBwhFiB,WAAy,QvB xhF7B,EuBwhFsC,iBvBxhFtC,EuBwhFyD,UvBxhFzD,EuBwhFqE,QvBxhFrE,C;QuByhFA,OAAO,W;O;KArBX, C;wFAwBA,yB;MvBxnFA,8C;MuBwnFA,kF;QAmB+D,iC;UAAA,oBAAYB,C;QAAG,0B;UAAA,aAAkB,C;QA AG,wB;UAAA,WAAgB,c;QvBvnF5H,UuBwnFA,iBvBxnFA,EuBwnFiB,WAAy,QvBxnF7B,EuBwnFsC,iBvBxn FtC,EuBwnFyD,UvBxnFzD,EuBwnFqE,QvBxnFrE,C;QuBynFA,OAAO,W;O;KArBX,C;wFAwBA,yB;MvBxnFA,8 C;MuBwnFA,kF;QAmBiE,iC;UAAA,oBAAYB,C;QAAG,0B;UAAA,aAAkB,C;QAAG,wB;UAAA,WAAgB,c;QvB vnF9H,UuBwnFA,iBvBxnFA,EuBwnFiB,WAAy,QvBxnF7B,EuBwnFsC,iBvBxnFtC,EuBwnFyD,UvBxnFzD,EuB wnFqE,QvBxnFrE,C;QuBynFA,OAAO,W;O;KArBX,C;kFAwBA,yB;MAAA,uC;MAAA,4B;QASI,OAAO,mBAA U,iBvB58EO,QuB48EjB,C;O;KATX,C;oFAYA,yB;MAAA,gD;MAAA,yC;MAAA,4B;QASI,OAAO,oBAAmB,O AAR,iBAAQ,CAAnB,C;O;KATX,C;oFAYA,yB;MAAA,yC;MAAA,4B;QASI,OAAO,oBAAW,iBvBx/EM,QuBw/ EjB,C;O;KATX,C;oFAYA,yB;MAAA,2C;MAAA,4B;QASI,OAAO,qBAAY,iBvB1/EK,QuB0/EjB,C;O;KATX,C; oFAYA,yB;MAAA,gD;MAAA,uC;MAAA,qC;QAWI,OAAO,mBAAkB,OAAR,iBAAQ,EAAO,OAAP,CAAlB,C; O;KAXX,C;oFAcA,yB;MAAA,gD;MAAA,yC;MAAA,qC;QAWI,OAAO,oBAAmB,OAAR,iBAAQ,EAAO,OAAP, CAAnB,C;O;KAXX,C;oFAcA,yB;MAAA,+C;MAAA,yC;MAAA,qC;QAWI,OAAO,oBAAmB,OAAR,iBAAQ,E AAO,OAAP,CAAnB,C;O;KAXX,C;oFAcA,yB;MAAA,gD;MAAA,2C;MAAA,qC;QAWI,OAAO,qBAAoB,OAAR,iBAAQ,EAAO,OAAP,CAApB,C;O;KAXX,C;4FAcA,yB;MAAA,0D;MAAA,uC;MAAA,gD;QAaI,OAAO,mBA AkB,YAAR,iBAAQ,EAAY,SAAZ,EAAuB,OAAvB,CAAlB,C;O;KAbX,C;8FAgBA,yB;MAAA,0D;MAAA,yC;M AAA,gD;QAaI,OAAO,oBAAmB,YAAR,iBAAQ,EAAY,SAAZ,EAAuB,OAAvB,CAAnB,C;O;KAbX,C;6FAgBA,yB;MAAA,0D;MAAA,2C;MAAA,gD;QAaI,OAAO,qBAAoB,YAAR,iBAAQ,EAAY, SAAZ,EAAuB,OAAvB,CAApB,C;O;KAbX,C;IAgBA,sD;MAWyC,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAA e,c;MACHE,OAAR,iBAAQ,EAAK,OnCv8GoB,KmCu8GzB,EAAAsB,SAAtB,EAAiC,OAAjC,C;K;IAGZ,wD;MA W2C,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,c;MACIE,OAAR,iBAAQ,EAAK,OnB38GsB,KmB28G3B,E AAuB,SAAvB,EAAC,OAAIC,C;K;IAGZ,wD;MAW2C,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,c;MACI E,OAAR,iBAAQ,EAAK,OpC7gHsB,KoC6gH3B,EAAuB,SAAvB,EAAC,OAAIC,C;K;IAGZ,wD;MAW6C,yB;Q AAA,YAAiB,C;MAAG,uB;QAAA,UAAe,c;MACpE,OAAR,iBAAQ,EAAK,OICjhHwB,KkCihH7B,EAAwB,SAA xB,EAAmC,OAAnC,C;K;8FASR,yB;MAAA,0D;MAAA,4B;QAAQ,OAAQ,YAAR,iBAAQ,C;O;KAAhB,C;8FAQ A,yB;MAAA,0D;MAAA,4B;QAAQ,OAAQ,YAAR,iBAAQ,C;O;KAAhB,C;+FAQA,yB;MAAA,0D;MAAA,4B;Q AAQ,OAAQ,YAAR,iBAAQ,C;O;KAAhB,C;+FAQA,yB;MAAA,0D;MAAA,4B;QAAQ,OAAQ,YAAR,iBAAQ,C; O;KAAhB,C;kGAQA,yB;MAAA,8D;MAAA,4B;QAAQ,OAAQ,cAAR,iBAAQ,C;O;KAAhB,C;kGAQA,yB;MAA

A,8D;MAAA,4B;QAAQ,OAAQ,cAAR,iBAAQ,C;O;KAAhB,C;mGAQA,yB;MAAA,8D;MAAA,4B;QAAQ,OAAQ,cAAR,iBAAQ,C;O;KAAhB,C;mGAQA,yB;MAAA,8D;MAAA,4B;QAAQ,OAAQ,cAAR,iBAAQ,C;O;KAAhB,C;iFAEJ,yB;MAAA,uC;MvBvoEA,iD;MuBuoEA,qC;QAOqB,4B;QAAA,gBAAU,OnC9jHM,K;QmC8jHjC,OAAO,mBvBzoEA,2BAxIK,gBAAW,SAAX,EAwIL,CuByoEA,C;O;KAPX,C;iFAUA,yB;MAAA,yC;MvBzoEA,iD;MuByoEA,qC;QAOI,OAAO,oBvB3oEA,qBuB2oEW,iBvB3oEX,EAxIK,mBuBmxEGb,OnB7jHO,KJ0yCvB,CAwIL,CuB2oEA,C;O;KAPX,C;iFAUA,yB;MAAA,yC;MvB3qEA,iD;MuB2qEA,qC;QAOsB,4B;QAAA,gBAAU,OpC1nHO,K;QoC0nHnC,OAAO,oBvB7qEA,2BAxIK,eAAY,SAAZ,EAwIL,CuB6qEA,C;O;KAPX,C;iFAUA,yB;MAAA,2C;MvB7qEA,iD;MuB6qEA,qC;QAOuB,4B;QAAA,gBAAU,OICznHQ,K;QkCynHrC,OAAO,qBvB/qEA,2BAxIK,gBAAa,SAAb,EAwIL,CuB+qEA,C;O;KAPX,C;IAUA,sC;MAQoB,UAAiB,M;MAFjC,YAAY,c;MACZ,aAAqB,UAAAR,iBAAQ,EAAO,iBAAO,QAAS,KAAhB,IAAP,C;MACL,0B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAU,OAAO,cAAP,EAAO,sBAAP,YAAkB,OnCvmHX,K;;MmCwmHjC,OAAO,cAAU,MAAV,C;K;IAGX,sC;MAQoB,UAAiB,M;MAFjC,YAAY,c;MACZ,aAAqB,UAAAR,iBAAQ,EAAO,iBAAO,QAAS,KAAhB,IAAP,C;MACL,0B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAU,OAAO,cAAP,EAAO,sBAAP,YAAkB,OnBxmHT,K;;MmBymHnC,OAAO,eAAW,MAAX,C;K;IAGX,sC;MAQoB,UAAiB,M;MAFjC,YAAY,c;MACZ,aAAqB,UAAAR,iBAAQ,EAAO,iBAAO,QAAS,KAAhB,IAAP,C;MACL,0B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAU,OAAO,cAAP,EAAO,sBAAP,YAAkB,OnCvqHT,K;;MoCwqHnC,OAAO,eAAW,MAAX,C;K;IAGX,sC;MAQoB,UAAiB,M;MAFjC,YAAY,c;MACZ,aAAqB,UAAAR,iBAAQ,EAAO,iBAAO,QAAS,KAAhB,IAAP,C;MACL,0B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAU,OAAO,cAAP,EAAO,sBAAP,YAAkB,OICxqHP,K;;MkCyqHrC,OAAO,gBAAY,MAAZ,C;K;iFAGX,yB;MAAA,uC;MvB/tEA,iD;MuB+tEA,sC;QAOI,OAAO,mBvBjuEA,qBuBiuEU,iBvBjuEV,EuBiuEoB,QAAS,QvBjuE7B,CuBiuEA,C;O;KAPX,C;iFAUA,yB;MAAA,yC;MvBjuEA,iD;MuBiuEA,sC;QAOI,OAAO,oBvBnuEA,qBuBmuEW,iBvBnuEX,EuBmuEqB,QAAS,QvBnuE9B,CuBmuEA,C;O;KAPX,C;iFAUA,yB;MAAA,yC;MvBnwEA,iD;MuBmwEA,sC;QAOI,OAAO,oBvBrwEA,qBuBqwEW,iBvBrwEX,EuBqwEqB,QAAS,QvBrwE9B,CuBqwEA,C;O;KAPX,C;iFAUA,yB;MAAA,2C;MvBrwEA,iD;MuBqwEA,sC;QAOI,OAAO,qBvBvwEA,qBuBuwEY,iBvBvwEZ,EuBuwEsB,QAAS,QvBvwE/B,CuBuwEA,C;O;KAPX,C;IAUA,2B;MAQI,IAAI,iBAAO,CAAX,C;QAAC,YAAU,SAAV,EAAGB,CAAhB,EAAMB,cAANB,C;K;IAGIB,2B;MAQI,IAAI,iBAAO,CAAX,C;QAAC,YAAU,SAAV,EAAGB,CAAhB,EAAMB,cAANB,C;K;IAGIB,2B;MAQI,IAAI,iBAAO,CAAX,C;QAAC,YAAU,SAAV,EAAGB,CAAhB,EAAMB,cAANB,C;K;IAGIB,+C;MAa0B,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,c;MACzD,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,cAAtC,C;MACb,YAAU,SAAV,EAAGB,SAAhB,EAA2B,OAA3B,C;K;IAGJ,+C;MAa2B,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,c;MAC1D,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,cAAtC,C;MACb,YAAU,SAAV,EAAGB,SAAhB,EAA2B,OAA3B,C;K;IAGJ,+C;MAa2B,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,c;MAC3D,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,cAAtC,C;MACb,YAAU,SAAV,EAAGB,SAAhB,EAA2B,OAA3B,C;K;IAGJ,0D;MAaI,kBAAK,SAAL,EAAGB,OAAhB,C;MAh8CQ,WAAR,iBAAQ,EAi8CA,SAj8CA,EAi8CW,OAj8CX,C;K;IAo8CZ,0D;MAaI,kBAAK,SAAL,EAAGB,OAAhB,C;MAj8CQ,WAAR,iBAAQ,EAk8CA,SAI8CA,EAk8CW,OAI8CX,C;K;IAq8CZ,0D;MAaI,kBAAK,SAAL,EAAGB,OAAhB,C;MAI8CQ,UAAAR,iBAAQ,EAm8CA,SAn8CA,EAm8CW,OAn8CX,C;K;IAS8CZ,0D;MAaI,kBAAK,SAAL,EAAGB,OAAhB,C;MAN8CQ,WAAR,iBAAQ,EAo8CA,SAP8CA,EAo8CW,OAP8CX,C;K;8FAu8CZ,qB;MAQI,OAAO,iBvB3jGiB,Q;K;4FuB8jG5B,qB;MAQI,OAAO,iBvBljGiB,Q;K;8FuBqjG5B,yB;MAAA,gD;MAAA,4B;QAQI,OAAe,OAAR,iBAAQ,C;O;KARnB,C;gGAWA,qB;MAQI,OAAO,iBvBlIgiB,Q;K;luB2lGL,gD;MAAA,wB;QAAW,qCAAK,KAAL,C;O;K;IANIC,iC;MAMI,OAAO,iBAAM,cAAN,EAAY,8BAAZ,C;K;IASY,kD;MAAA,wB;QAAW,qCAAK,KAAL,C;O;K;IANIC,mC;MAMI,OAAO,iBAAM,cAAN,EAAY,gCAAZ,C;K;IASY,kD;MAAA,wB;QAAW,qCAAK,KAAL,C;O;K;IANIC,mC;MAMI,OAAO,iBAAM,cAAN,EAAY,gCAAZ,C;K;IASY,kD;MAAA,wB;QAAW,qCAAK,KAAL,C;O;K;IANIC,mC;MAMI,OAAO,iBAAM,cAAN,EAAY,gCAAZ,C;K;IASiB,gD;MAAA,wB;QAAW,yBAAK,KAAL,C;O;K;IANvC,iC;MAMI,OJnqIO,eAAW,+BIImqIA,gBJnqIA,GAAGB,kBIImqIV,8BJnqIU,CAAhB,C,AAX,C;K;gGIsqIX,yB;MAAA,yC;MAAA,4B;QAQI,OAAO,oBAAW,SvBppGM,QuBopGjB,C;O;KARX,C;IAiB2B,8C;MAAA,wB;QAAW,wBAAK,KAAL,C;O;K;IANtC,gC;MAMI,OHvrIO,cAAU,gCGurIA,gBHvrIA,GAAe,iB

GurIT,6BHvrIS,CAAf,CAAV,C;K;8FG0rIX,yB;MAAA,uC;MAAA,4B;QAQI,OAAO,mBAAU,SvBppGO,QuBop GJb,C;O;KARX,C;IAiB4B,gD;MAAA,wB;QAAW,yBAAK,KAAL,C;O;K;IANvC,iC;MAMI,OF3sIO,eAAW,kBE 2sIA,gBF3sIA,EAAgB,kBE2sIV,8BF3sIU,CAAhB,CAAX,C;K;gGE8sIX,yB;MAAA,gD;MAAA,yC;MAAA,4B;Q AQI,OAAO,oBAAgB,OAAL,SAAK,CAAhB,C;O;KARX,C;IAiB6B,kD;MAAA,wB;QAAW,0BAAK,KAAL,C;O; K;IANxC,kC;MAMI,OD/tIO,gBAAY,gCC+tIA,gBD/tIA,GAAiB,mBC+tIX,+BD/tIW,CAAjB,CAAZ,C;K;kGCkuI X,yB;MAAA,2C;MAAA,4B;QAQI,OAAO,qBAAY,SvBtsGK,QuBssGjB,C;O;KARX,C;mGAWA,yB;MAAA,0D; MAAA,yD;MAAA,uE;MAAA,2C;QAcI,aAAa,mBAAyC,cAAIB,YAAY,cAAZ,CAAKB,EAAc,EAAAd,CAAzC,C;Q AsEG,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UArEO,MAsEP,aAAI,OAAJ,EAtEe,aAsEF,CAAc,O AAd,CAAb,C;;QAtEhB,OAAuB,M;O;Kaf3B,C;mGAKBA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,2C;QAc I,aAAa,mBAA0C,cAAIB,YAAY,cAAZ,CAAKB,EAAc,EAAAd,CAA1C,C;QAsEG,Q;QAAA,2B;QAAhB,OAAGB,c AAhB,C;UAAgB,yB;UArEO,MAsEP,aAAI,OAAJ,EAtEe,aAsEF,CAAc,OAAAd,CAAb,C;;QAtEhB,OAAuB,M;O; Kaf3B,C;kGAKBA,yB;MAAA,0D;MAAA,yD;MAAA,uE;MAAA,2C;QAcI,aAAa,mBAA0C,cAAIB,YAAY,cAA Z,CAAKB,EAAc,EAAAd,CAA1C,C;QAsEG,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UArEO,MAsEP, aAAI,OAAJ,EAtEe,aAsEF,CAAc,OAAAd,CAAb,C;;QAtEhB,OAAuB,M;O;Kaf3B,C;mGAKBA,yB;MAAA,0D;M AAA,yD;MAAA,uE;MAAA,2C;QAcI,aAAa,mBAA2C,cAAIB,YAAY,cAAZ,CAAKB,EAAc,EAAAd,CAA3C,C;QA sEG,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UArEO,MAsEP,aAAI,OAAJ,EAtEe,aAsEF,CAAc,OA Ad,CAAb,C;;QAtEhB,OAAuB,M;O;Kaf3B,C;uGAKBA,iD;MAYoB,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;Q AAgB,yB;QACZ,WAAy,aAAI,OAAJ,EAAa,cAAc,OAAAd,CAAb,C;;MAEhB,OAAO,W;K;uGAGX,iD;MAYoB,Q ;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAgB,yB;QACZ,WAAy,aAAI,OAAJ,EAAa,cAAc,OAAAd,CAAb,C;;M AEhB,OAAO,W;K;uGAGX,iD;MAYoB,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAgB,yB;QACZ,WAAy,aA AI,OAAJ,EAAa,cAAc,OAAAd,CAAb,C;;MAEhB,OAAO,W;K;uGAGX,iD;MAYoB,Q;MAAA,2B;MAAhB,OAAG B,cAAhB,C;QAAgB,yB;QACZ,WAAy,aAAI,OAAJ,EAAa,cAAc,OAAAd,CAAb,C;;MAEhB,OAAO,W;K;uFAGX, yB;MAAA,+D;MAoLA,gD;MApLA,uC;QASW,kBAAU,gB;QAKLD,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;U AAgB,yB;UACZ,WAnL6B,SAmLIB,CAAU,OAAV,C;UACC,OAAZ,WAAy,EAAO,IAAP,C;;QApLhB,OAsLO, W;O;KA/LX,C;uFAYA,yB;MAAA,+D;MAsLA,gD;MAtLA,uC;QASW,kBAAU,gB;QAoLD,Q;QAAA,2B;QAAh B,OAAGB,cAAhB,C;UAAgB,yB;UACZ,WArL6B,SAqLIB,CAAU,OAAV,C;UACC,OAAZ,WAAy,EAAO,IAAP, C;;QAtLhB,OAwLO,W;O;KAjMX,C;uFAYA,yB;MAAA,+D;MAwLA,gD;MAxLA,uC;QASW,kBAAU,gB;QAsL D,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,WAvL6B,SAuLIB,CAAU,OAAV,C;UACC,OAAZ ,WAAy,EAAO,IAAP,C;;QAxLhB,OA0LO,W;O;KAnMX,C;uFAYA,yB;MAAA,+D;MA0LA,gD;MA1LA,uC;QA SW,kBAAU,gB;QAwLD,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,WAzL6B,SAYLIB,CAAU, OAAV,C;UACC,OAAZ,WAAy,EAAO,IAAP,C;;QA1LhB,OA4LO,W;O;KArMX,C;qGAYA,yB;MAAA,+D;MA4 DA,gD;MA5DA,uC;QAYW,kBAAiB,gB;QA2DR,gB;QADhB,YAAY,C;QACI,2B;QAAhB,OAAGB,cAAhB,C;UA AgB,yB;UACZ,WA5DoC,SA4DzB,EAAU,cAAV,EAAU,sBAAV,WAAmB,OAAAnB,C;UACC,OAAZ,WAAy,EA AO,IAAP,C;;QA7DhB,OA+DO,W;O;KA3EX,C;qGAeA,yB;MAAA,+D;MA+DA,gD;MA/DA,uC;QAYW,kBAAi B,gB;QA8DR,gB;QADhB,YAAY,C;QACI,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,WA/DoC,SA+DzB, EAAU,cAAV,EAAU,sBAAV,WAAmB,OAAAnB,C;UACC,OAAZ,WAAy,EAAO,IAAP,C;;QAhEhB,OAKEO,W;O ;KA9EX,C;qGAeA,yB;MAAA,+D;MAkEA,gD;MAIEA,uC;QAYW,kBAAiB,gB;QAiER,gB;QADhB,YAAY,C;Q ACI,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,WAlEoC,SAkEzB,EAAU,cAAV,EAAU,sBAAV,WAAmB ,OAAAnB,C;UACC,OAAZ,WAAy,EAAO,IAAP,C;;QAnEhB,OaqEO,W;O;KAjFX,C;qGAeA,yB;MAAA,+D;MAq EA,gD;MArEA,uC;QAYW,kBAAiB,gB;QAoER,gB;QADhB,YAAY,C;QACI,2B;QAAhB,OAAGB,cAAhB,C;UA AgB,yB;UACZ,WArEoC,SAqEzB,EAAU,cAAV,EAAU,sBAAV,WAAmB,OAAAnB,C;UACC,OAAZ,WAAy,EA AO,IAAP,C;;QAtEhB,OAwEO,W;O;KApFX,C;yGAeA,yB;MAAA,gD;MAAA,oD;QAWoB,UACS,M;QAFzB,YA AY,C;QACI,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,WAAW,WAAU,cAAV,EAAU,sBAAV,WAAmB, OAAAnB,C;UACC,OAAZ,WAAy,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KafX,C;yGAKBA,yB;MAAA,gD;MAAA ,oD;QAWoB,UACS,M;QAFzB,YAAY,C;QACI,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,WAAW,WAA U,cAAV,EAAU,sBAAV,WAAmB,OAAAnB,C;UACC,OAAZ,WAAy,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KafX, C;yGAKBA,yB;MAAA,gD;MAAA,oD;QAWoB,UACS,M;QAFzB,YAAY,C;QACI,2B;QAAhB,OAAGB,cAAhB,C ;UAAgB,yB;UACZ,WAAW,WAAU,cAAV,EAAU,sBAAV,WAAmB,OAAAnB,C;UACC,OAAZ,WAAy,EAAO,IA

AP,C;;QAEhB,OAAO,W;O;KafX,C;yGakBA,yB;MAAA,gD;MAAA,oD;QAWoB,UACS,M;QAFzB,YAAY,C;Q  
ACI,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAW,WAAU,cAAV,EAAU,sBAAV,WAAmB,OAAhB  
,C;UACC,OAAZ,WAA Y,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KafX,C;2FAkBA,yB;MAAA,gD;MAAA,oD;QA  
OoB,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAW,UAAU,OAAV,C;UACC,OAAZ,WAA  
Y,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KAXX,C;2FAcA,yB;MAAA,gD;MAAA,oD;QAOoB,Q;QAAA,2B;QAA  
hB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAAW,UAAU,OAAV,C;UACC,OAAZ,WAA Y,EAAO,IAAP,C;;QAE  
hB,OAAO,W;O;KAXX,C;2FAcA,yB;MAAA,gD;MAAA,oD;QAOoB,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;U  
AAgB,yB;UACZ,WAAW,UAAU,OAAV,C;UACC,OAAZ,WAA Y,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KAXX,  
C;2FAcA,yB;MAAA,gD;MAAA,oD;QAOoB,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,WAA  
W,UAAU,OAAV,C;UACC,OAAZ,WAA Y,EAAO,IAAP,C;;QAEhB,OAAO,W;O;KAXX,C;uFAcA,yB;MAAA,w  
E;MA4HA,+D;MA5HA,yC;QAYW,kBAAU,oB;QA4HD,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;U  
ACZ,UA7HoD,WA6H1C,CAAY,OAAZ,C;U/B59IP,U;UADP,Y+B89Ie,W/B99IH,W+B89IwB,G/B99Ix B,C;UAC  
L,IAAI,aAAJ,C;YACH,a+B49IuC,gB;YAA5B,W/B39IX,a+B29IgC,G/B39IhC,EAAS,MAAT,C;YACA,e;;YAEA,  
c;;U+Bw9IA,iB;UACA,IAAK,WAAI,OAAJ,C;;QA/HT,OaiIo,W;O;KA7IX,C;uFAeA,yB;MAAA,wE;MAiIA,+D;  
MAjIA,yC;QAYW,kBAAU,oB;QaiID,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,UAlIqD,Wak  
I3C,CAAY,OAAZ,C;U/Bh/IP,U;UADP,Y+Bk/Ie,W/BI/IH,W+Bk/IwB,G/BI/IxB,C;UACL,IAAI,aAAJ,C;YACH,a+  
Bg/IuC,gB;YAA5B,W/B/+IX,a+B++IgC,G/B/+IhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;U+B4+IA,iB;UACA,IAA  
K,WAAI,OAAJ,C;;QApIT,OAsIO,W;O;KAlJX,C;sFAeA,yB;MAAA,wE;MAsIA,+D;MatIA,yC;QAYW,kBAAU,  
oB;QAsID,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,UAvIqD,WauI3C,CAAY,OAAZ,C;U/Bp  
gJP,U;UADP,Y+BsgJe,W/BtgJH,W+BsgJwB,G/BtgJxB,C;UACL,IAAI,aAAJ,C;YACH,a+BogJuC,gB;YAA5B,W/  
BngJX,a+BmgJgC,G/BngJhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;U+BggJA,iB;UACA,IAAK,WAAI,OAAJ,C;;Q  
AzIT,OA2IO,W;O;KA vJX,C;uFAeA,yB;MAAA,wE;MA2IA,+D;MA3IA,yC;QAYW,kBAAU,oB;QA2ID,Q;QAA  
A,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,UA5IsD,WA4I5C,CAAY,OAAZ,C;U/BxhJP,U;UADP,Y+B0  
hJe,W/B1hJH,W+B0hJwB,G/B1hJxB,C;UACL,IAAI,aAAJ,C;YACH,a+BwhJuC,gB;YAA5B,W/BvhJX,a+BuhJgC  
,G/BvhJhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;U+BohJA,iB;UACA,IAAK,WAAI,OAAJ,C;;QA9IT,OAqJO,W;O  
;KA5JX,C;uFAeA,yB;MAAA,wE;MAgJA,+D;MAhJA,yD;QAaW,kBAAU,oB;QAqJD,Q;QAAA,2B;QAAhB,OA  
AgB,cAAhB,C;UAAgB,yB;UACZ,UajJiD,WaiJvC,CAAY,OAAZ,C;U/B7iJP,U;UADP,Y+B+iJe,W/B/iJH,W+B+  
iJwB,G/B/iJxB,C;UACL,IAAI,aAAJ,C;YACH,a+B6iJuC,gB;YAA5B,W/B5iJX,a+B4iJgC,G/B5iJhC,EAAS,MAAT  
,C;YACA,e;;YAEA,c;;U+ByiJA,iB;UACA,IAAK,WAnJyD,cAmJrD,CAAe,OAaf,CAAJ,C;;QAnJT,OAqJO,W;O;  
KAIKX,C;uFAgBA,yB;MAAA,wE;MAqJA,+D;MARJA,yD;QAaW,kBAAU,oB;QAqJD,Q;QAAA,2B;QAAhB,OA  
AgB,cAAhB,C;UAAgB,yB;UACZ,UAtJiD,WAsJvC,CAAY,OAAZ,C;U/BlkJP,U;UADP,Y+BokJe,W/BpkJH,W+B  
okJwB,G/BpkJxB,C;UACL,IAAI,aAAJ,C;YACH,a+BkkJuC,gB;YAA5B,W/BjkJX,a+BikJgC,G/BjkJhC,EAAS,M  
AAT,C;YACA,e;;YAEA,c;;U+B8jJA,iB;UACA,IAAK,WaxJyD,cAwJrD,CAAe,OAaf,CAAJ,C;;QAxJT,OA0JO,  
W;O;KA vKX,C;uFAgBA,yB;MAAA,wE;MA0JA,+D;MA1JA,yD;QAaW,kBAAU,oB;QA0JD,Q;QAAA,2B;QAAh  
B,OAAgB,cAAhB,C;UAAgB,yB;UACZ,UA3JiD,WA2JvC,CAAY,OAAZ,C;U/BvlJP,U;UADP,Y+BvlJe,W/BzlJH,  
W+BvlJwB,G/BzlJxB,C;UACL,IAAI,aAAJ,C;YACH,a+BulJuC,gB;YAA5B,W/BtlJX,a+BslJgC,G/BtlJhC,EAAS,  
MAAT,C;YACA,e;;YAEA,c;;U+BmlJA,iB;UACA,IAAK,WA7JyD,cA6JrD,CAAe,OAaf,CAAJ,C;;QA7JT,OA+J  
O,W;O;KA5KX,C;uFAgBA,yB;MAAA,wE;MA+JA,+D;MA/JA,yD;QAaW,kBAAU,oB;QA+JD,Q;QAAA,2B;QA  
AhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,UAhKiD,WAgKvC,CAAY,OAAZ,C;U/B5mJP,U;UADP,Y+B8mJe,W/  
/B9mJH,W+B8mJwB,G/B9mJxB,C;UACL,IAAI,aAAJ,C;YACH,a+B4mJuC,gB;YAA5B,W/B3mJX,a+B2mJgC,G/  
B3mJhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;U+BwmJA,iB;UACA,IAAK,WAlKyD,cAkKrD,CAAe,OAaf,CAAJ  
,C;;QAIKT,OAoKO,W;O;KAjLX,C;2FAgBA,yB;MAAA,+D;MAAA,sD;QAYoB,Q;QAAA,2B;QAAhB,OAAgB,c  
AAhB,C;UAAgB,yB;UACZ,UAAU,YAAY,OAAZ,C;U/B59IP,U;UADP,Y+B89Ie,W/B99IH,W+B89IwB,G/B99Ix  
B,C;UACL,IAAI,aAAJ,C;YACH,a+B49IuC,gB;YAA5B,W/B39IX,a+B29IgC,G/B39IhC,EAAS,MAAT,C;YACA,e  
;;YAEA,c;;U+Bw9IA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAET,OAAO,W;O;KAjBX,C;2FAoBA,yB;MAAA,+D;  
MAAA,sD;QAYoB,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,UAAU,YAAY,OAAZ,C;U/Bh/I  
P,U;UADP,Y+Bk/Ie,W/BI/IH,W+Bk/IwB,G/BI/IxB,C;UACL,IAAI,aAAJ,C;YACH,a+Bg/IuC,gB;YAA5B,W/B/+I  
X,a+B++IgC,G/B/+IhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;U+B4+IA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAET,

OAAO,W;O;KAjBX,C;2FAoBA,yB;MAAA,+D;MAAA,sD;QAYoB,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UA  
AgB,yB;UACZ,UAAU,YAAY,OAAZ,C;U/BpgJP,U;UADP,Y+BsgJe,W/BtgJH,W+BsgJwB,G/BtgJxB,C;UACL,I  
AAI,aAAJ,C;YACH,a+BogJuC,gB;YAA5B,W/BngJX,a+BmgJgC,G/BngJhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;  
U+BggJA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAET,OAAO,W;O;KAjBX,C;2FAoBA,yB;MAAA,+D;MAAA,sD;Q  
AYoB,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,UAAU,YAAY,OAAZ,C;U/BxhJP,U;UADP,Y  
+B0hJe,W/B1hJH,W+B0hJwB,G/B1hJxB,C;UACL,IAAI,aAAJ,C;YACH,a+BwhJuC,gB;YAA5B,W/BvhJX,a+Buh  
JgC,G/BvhJhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;U+BohJA,iB;UACA,IAAK,WAAI,OAAJ,C;;QAET,OAAO,W  
;O;KAjBX,C;2FAoBA,yB;MAAA,+D;MAAA,sE;QAaoB,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;U  
ACZ,UAAU,YAAY,OAAZ,C;U/B7iJP,U;UADP,Y+B+iJe,W/B/iJH,W+B+iJwB,G/B/iJxB,C;UACL,IAAI,aAAJ,C;  
YACH,a+B6iJuC,gB;YAA5B,W/B5iJX,a+B4iJgC,G/B5iJhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;U+ByiJA,iB;UA  
CA,IAAK,WAAI,eAAe,OAaf,CAAJ,C;;QAET,OAAO,W;O;KAIBX,C;2FAqBA,yB;MAAA,+D;MAAA,sE;QAao  
B,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,UAAU,YAAY,OAAZ,C;U/BikJP,U;UADP,Y+Bok  
Je,W/BpkJH,W+BokJwB,G/BpkJxB,C;UACL,IAAI,aAAJ,C;YACH,a+BkkJuC,gB;YAA5B,W/BjkJX,a+BikJgC,G/  
BjkJhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;U+B8jJA,iB;UACA,IAAK,WAAI,eAAe,OAaf,CAAJ,C;;QAET,OA  
AO,W;O;KAIBX,C;2FAqBA,yB;MAAA,+D;MAAA,sE;QAaoB,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB  
,yB;UACZ,UAAU,YAAY,OAAZ,C;U/BvIJP,U;UADP,Y+BvIJe,W/BzIjH,W+BvIjwB,G/BzIjxB,C;UACL,IAAI,a  
AAJ,C;YACH,a+BulJuC,gB;YAA5B,W/BtlJX,a+BslJgC,G/BtlJhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;U+BmlJA  
,iB;UACA,IAAK,WAAI,eAAe,OAaf,CAAJ,C;;QAET,OAAO,W;O;KAIBX,C;2FAqBA,yB;MAAA,+D;MAAA,sE  
;QAaoB,Q;QAAA,2B;QAAhB,OAAGB,cAAhB,C;UAAgB,yB;UACZ,UAAU,YAAY,OAAZ,C;U/B5mJP,U;UADP  
,Y+B8mJe,W/B9mJH,W+B8mJwB,G/B9mJxB,C;UACL,IAAI,aAAJ,C;YACH,a+B4mJuC,gB;YAA5B,W/B3mJX,  
a+B2mJgC,G/B3mJhC,EAAS,MAAT,C;YACA,e;;YAEA,c;;U+BwmJA,iB;UACA,IAAK,WAAI,eAAe,OAaf,CA  
AJ,C;;QAET,OAAO,W;O;KAIBX,C;+EAqBA,yB;MAAA,gE;MAAA,uC;QAUW,kBAAM,eAAa,cAAb,C;QAsKA  
,Q;QAAA,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,WAAY,WAvKiB,SAuKb,CAAU,IAAV,CAAJ,C;;QAvKhB  
,OAwKO,W;O;KAILX,C;+EAaA,yB;MAAA,gE;MAAA,uC;QAUW,kBAAM,eAAa,cAAb,C;QAsKA,Q;QAAA,2  
B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,WAAY,WAvKiB,SAuKb,CAAU,IAAV,CAAJ,C;;QAvKhB,OAwKO,  
W;O;KAILX,C;8EAaA,yB;MAAA,gE;MAAA,uC;QAUW,kBAAM,eAAa,cAAb,C;QAsKA,Q;QAAA,2B;QAAb,O  
AAa,cAAb,C;UAAa,sB;UACT,WAAY,WAvKiB,SAuKb,CAAU,IAAV,CAAJ,C;;QAvKhB,OAwKO,W;O;KAILX  
,C;+EAaA,yB;MAAA,gE;MAAA,uC;QAUW,kBAAM,eAAa,cAAb,C;QAsKA,Q;QAAA,2B;QAAb,OAAa,cAAb,  
C;UAAa,sB;UACT,WAAY,WAvKiB,SAuKb,CAAU,IAAV,CAAJ,C;;QAvKhB,OAwKO,W;O;KAILX,C;4FAaA,y  
B;MAAA,gE;MAAA,uC;QAUW,kBAaA,eAAa,cAAb,C;QAqDP,gB;QADb,YAAY,C;QACC,2B;QAAb,OAAa,cA  
Ab,C;UAAa,sB;UACT,WAAY,WAtDwB,SAsDpB,EAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;QAt  
DhB,OAuDO,W;O;KAjEX,C;6FAaA,yB;MAAA,gE;MAAA,uC;QAUW,kBAaA,eAAa,cAAb,C;QAWDP,gB;QAD  
b,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,WAAY,WAZDwB,SAYDpB,EAAU,cAAV,EAAU,  
sBAAV,WAAmB,IAAnB,CAAJ,C;;QAZDhB,OA0DO,W;O;KApEX,C;6FAaA,yB;MAAA,gE;MAAA,uC;QAUW,  
kBAaA,eAAa,cAAb,C;QA2DP,gB;QADb,YAAY,C;QACC,2B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,WAAY,W  
A5DwB,SA4DpB,EAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;QA5DhB,OA6DO,W;O;KAvEX,C;4F  
AaA,yB;MAAA,gE;MAAA,uC;QAUW,kBAaA,eAAa,cAAb,C;QA8DP,gB;QADb,YAAY,C;QACC,2B;QAAb,OA  
Aa,cAAb,C;UAAa,sB;UACT,WAAY,WA/DwB,SA+DpB,EAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,  
C;;QA/DhB,OAgEO,W;O;KA1EX,C;iGAaA,6C;MAWiB,UACiB,M;MAF9B,YAAY,C;MACC,2B;MAAb,OAAa,  
cAAb,C;QAAa,sB;QACT,WAAY,WAAI,WAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;MACHB,OA  
AO,W;K;iGAGX,6C;MAWiB,UACiB,M;MAF9B,YAAY,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,  
WAAY,WAAI,WAAU,cAAV,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;MACHB,OAAO,W;K;iGAGX,6C;MAW  
iB,UACiB,M;MAF9B,YAAY,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,WAAY,WAAI,WAAU,cAA  
V,EAAU,sBAAV,WAAmB,IAAnB,CAAJ,C;;MACHB,OAAO,W;K;iGAGX,6C;MAWiB,UACiB,M;MAF9B,YAA  
Y,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,WAAY,WAAI,WAAU,cAAV,EAAU,sBAAV,WAAmB,I  
AAnB,CAAJ,C;;MACHB,OAAO,W;K;mFAGX,6C;MAQiB,Q;MAAA,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QAC  
T,WAAY,WAAI,UAAU,IAAV,CAAJ,C;;MACHB,OAAO,W;K;mFAGX,6C;MAQiB,Q;MAAA,2B;MAAb,OAAa,  
cAAb,C;QAAa,sB;QACT,WAAY,WAAI,UAAU,IAAV,CAAJ,C;;MACHB,OAAO,W;K;mFAGX,6C;MAQiB,Q;M

AAA,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,WAAY,WAAI,UAAU,IAAV,CAAJ,C;;MACHb,OAAO,W;K;m  
FAGX,6C;MAQIB,Q;MAAA,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,WAAY,WAAI,UAAU,IAAV,CAAJ,C;;  
MACHb,OAAO,W;K;IAUiB,6C;MAAA,mB;QAAE,gC;O;K;IAP9B,iC;MAOI,OAAO,qBAAiB,8BAAjB,C;K;IAU  
iB,6C;MAAA,mB;QAAE,gC;O;K;IAP9B,iC;MAOI,OAAO,qBAAiB,8BAAjB,C;K;IAUiB,6C;MAAA,mB;QAAE,  
gC;O;K;IAP9B,iC;MAOI,OAAO,qBAAiB,8BAAjB,C;K;IAUiB,6C;MAAA,mB;QAAE,gC;O;K;IAP9B,iC;MAOI,  
OAAO,qBAAiB,8BAAjB,C;K;+EAGX,gC;MASoB,Q;MAAA,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QAAM  
,IAAI,CAAC,UAAU,OAAV,CAAL,C;UAAyB,OAAO,K;;MACtD,OAAO,I;K;+EAGX,gC;MASoB,Q;MAAA,2B;  
MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QAAM,IAAI,CAAC,UAAU,OAAV,CAAL,C;UAAyB,OAAO,K;;MACtD  
,OAAO,I;K;+EAGX,gC;MASoB,Q;MAAA,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QAAM,IAAI,CAAC,UA  
AU,OAAV,CAAL,C;UAAyB,OAAO,K;;MACtD,OAAO,I;K;+EAGX,gC;MASoB,Q;MAAA,2B;MAAhB,OAAGb,  
cAAhB,C;QAAGb,yB;QAAM,IAAI,CAAC,UAAU,OAAV,CAAL,C;UAAyB,OAAO,K;;MACtD,OAAO,I;K;+EA  
GX,yB;MAAA,0C;MAAA,4B;QASI,OAAe,IAAR,iBAAQ,C;O;KATnB,C;+EAYA,yB;MAAA,0C;MAAA,4B;QA  
SI,OAAe,IAAR,iBAAQ,C;O;KATnB,C;+EAYA,yB;MAAA,0C;MAAA,4B;QASI,OAAe,IAAR,iBAAQ,C;O;KAT  
nB,C;+EAYA,yB;MAAA,0C;MAAA,4B;QASI,OAAe,IAAR,iBAAQ,C;O;KATnB,C;+EAYA,gC;MASoB,Q;MAA  
A,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,I;;MACrD,  
OAAO,K;K;+EAGX,gC;MASoB,Q;MAAA,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QAAM,IAAI,UAAU,OA  
AV,CAAJ,C;UAAwB,OAAO,I;;MACrD,OAAO,K;K;+EAGX,gC;MASoB,Q;MAAA,2B;MAAhB,OAAGb,cAAhB  
,C;QAAGb,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,I;;MACrD,OAAO,K;K;+EAGX,gC;MASoB  
,Q;MAAA,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,I;;  
MACrD,OAAO,K;K;mFAGX,gC;MAQoB,Q;MADhB,YAAY,C;MACI,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,y  
B;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,qB;;MAC9C,OAAO,K;K;mFAGX,gC;MAQoB,Q;MADhB,YA  
AY,C;MACI,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,qB;;MA  
C9C,OAAO,K;K;mFAGX,gC;MAQoB,Q;MADhB,YAAY,C;MACI,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;Q  
AAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,qB;;MAC9C,OAAO,K;K;mFAGX,gC;MAQoB,Q;MADhB,YAAY,C  
;MACI,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,qB;;MAC9C,O  
AAO,K;K;iFAGX,yC;MAaoB,Q;MADhB,kBAakB,O;MACF,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QAAM  
,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;;MACpC,OAAO,W;K;iFAGX,yC;MAaoB,Q;MADhB,kBAakB,O;MA  
CF,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QAAM,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;;MACpC,OAAO  
,W;K;iFAGX,yC;MAaoB,Q;MADhB,kBAakB,O;MACF,2B;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QAAM,cAA  
c,UAAU,WAAV,EAAuB,OAAvB,C;;MACpC,OAAO,W;K;iFAGX,yC;MAaoB,Q;MADhB,kBAakB,O;MACF,2B  
;MAAhB,OAAGb,cAAhB,C;QAAGb,yB;QAAM,cAAc,UAAU,WAAV,EAAuB,OAAvB,C;;MACpC,OAAO,W;K;  
+FAGX,yC;MAeoB,UAA8B,M;MAF9C,YAAY,C;MACZ,kBAakB,O;MACF,2B;MAAhB,OAAGb,cAAhB,C;QA  
AGb,yB;QAAM,cAAc,WAAU,cAAV,EAAU,sBAAV,WAAmB,WAAAnB,EAAGC,OAAhC,C;;MACpC,OAAO,W;  
K;+FAGX,yC;MAeoB,UAA8B,M;MAF9C,YAAY,C;MACZ,kBAakB,O;MACF,2B;MAAhB,OAAGb,cAAhB,C;  
QAAGb,yB;QAAM,cAAc,WAAU,cAAV,EAAU,sBAAV,WAAmB,WAAAnB,EAAGC,OAAhC,C;;MACpC,OAAO,  
W;K;+FAGX,yC;MAeoB,UAA8B,M;MAF9C,YAAY,C;MACZ,kBAakB,O;MACF,2B;MAAhB,OAAGb,cAAhB,  
C;QAAGb,yB;QAAM,cAAc,WAAU,cAAV,EAAU,sBAAV,WAAmB,WAAAnB,EAAGC,OAAhC,C;;MACpC,OAA  
O,W;K;+FAGX,yC;MAeoB,UAA8B,M;MAF9C,YAAY,C;MACZ,kBAakB,O;MACF,2B;MAAhB,OAAGb,cAAh  
B,C;QAAGb,yB;QAAM,cAAc,WAAU,cAAV,EAAU,sBAAV,WAAmB,WAAAnB,EAAGC,OAAhC,C;;MACpC,OA  
AO,W;K;0FAGX,yB;MA1uDI,8D;MA0uDJ,gD;QAeoC,Q;QAHhC,YAtvDgB,cAAR,iBAAQ,C;QAuvDhB,kBA  
AkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,uBAAI,YAAJ,EAAL,oBAAJ,QAAV,EAAwB,WAA  
xB,C;;QAEIB,OAAO,W;O;KAjBX,C;2FAoBA,yB;MATvDI,8D;MASvDJ,gD;QAeoC,Q;QAHhC,YAlwDgB,cAAR,  
iBAAQ,C;QAmwDhB,kBAakB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,uBAAI,YAAJ,EAAL,oB  
AAJ,QAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KAjBX,C;2FAoBA,yB;MA1wDI,8D;MAkwDJ,gD;QAeoC,  
Q;QAHhC,YA9wDgB,cAAR,iBAAQ,C;QA+wDhB,kBAakB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,U  
AAU,uBAAI,YAAJ,EAAL,oBAAJ,QAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KAjBX,C;2FAoBA,yB;MA9  
wDI,8D;MA8wDJ,gD;QAeoC,Q;QAHhC,YA1xDgB,cAAR,iBAAQ,C;QA2xDhB,kBAakB,O;QACIB,OAAO,SA  
AS,CAAhB,C;UACI,cAAc,UAAU,uBAAI,YAAJ,EAAL,oBAAJ,QAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;

KAjBX,C;yGAoBA,yB;MA1zDI,8D;MA0zDJ,gD;QAaI,YAv0DgB,cAAR,iBAAQ,C;QAw0DhB,kBAAkB,O;QAC  
IB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,sBAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,  
qB;;QAEJ,OAAO,W;O;KAnBX,C;yGAsBA,yB;MAx0DI,8D;MAw0DJ,gD;QAaI,YAr1DgB,cAAR,iBAAQ,C;QAs  
1DhB,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,sBAAI,KAAJ,CAAjB,EA  
A6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAnBX,C;yGAsBA,yB;MA1DI,8D;MA1DJ,gD;QAaI,YAn2Dg  
B,cAAR,iBAAQ,C;QAo2DhB,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,s  
BAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAnBX,C;yGAsBA,yB;MAP2DI,8D;MA  
o2DJ,gD;QAaI,YAj3DgB,cAAR,iBAAQ,C;Qak3DhB,kBAAkB,O;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,  
UAAU,KAAV,EAAiB,sBAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAnBX,C;uFAs  
BA,6B;MAOoB,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,OAAO,OAAP,C;;K;uFAG1B,6B;  
MAOoB,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,OAAO,OAAP,C;;K;uFAG1B,6B;MAOoB,Q;MAA  
A,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,OAAO,OAAP,C;;K;qGAG1B,6B;MAUiB,UAAa,M;MAD1  
B,YAAY,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QAAM,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;  
;K;qGAGvB,6B;MAUiB,UAAa,M;MAD1B,YAAY,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QAAM,QAAO  
,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;K;qGAGvB,6B;MAUiB,UAAa,M;MAD1B,YAAY,C;MACC,2B;MAA  
b,OAAa,cAAb,C;QAAa,sB;QAAM,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;K;qGAGvB,6B;MAUiB,UA  
Aa,M;MAD1B,YAAY,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QAAM,QAAO,cAAP,EAAO,sBAAP,WAA  
gB,IAAhB,C;;K;IAGvB,2B;MAKI,OAAO,uB;K;IAGX,2B;MAKI,OAAO,uB;K;IAGX,2B;MAKI,OAAO,uB;K;IA  
GX,2B;MAKI,OAAO,uB;K;mFAGX,yB;MA9gEI,8D;MA8gEJ,sC;QAMW,sB;;UAuCP,IAAI,mBAAJ,C;YAAe,qB  
AAO,I;YAAP,uB;WACf,cAAc,sBAAK,CAAL,C;UACd,gBA7jEgB,cAAR,iBAAQ,C;UA8jEhB,IAAI,cAAa,CAAj  
B,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eA3CmB,QA2CJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,  
M;YACI,QAAQ,sBAAK,CAAL,C;YACR,QA9Ce,QA8CP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C  
;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QApDP,yB;O;KANJ,C;mFASA,yB;MA+gEJ,8D;MA+gEJ  
,sC;QAMW,sB;;UAuDP,IAAI,mBAAJ,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,sBAAK,CAAL,C;UACd,gBA9  
kEgB,cAAR,iBAAQ,C;UA+kEhB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eA3DmB,QA2D  
J,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,sBAAK,CAAL,C;YACR,QA9De,QA8DP,C  
AAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QA  
pEP,yB;O;KANJ,C;mFASA,yB;MAhhEI,8D;MAghEJ,sC;QAMW,sB;;UAuEP,IAAI,mBAAJ,C;YAAe,qBAAO,I;Y  
AAP,uB;WACf,cAAc,sBAAK,CAAL,C;UACd,gBA/IEgB,cAAR,iBAAQ,C;UAgmEhB,IAAI,cAAa,CAAjB,C;YA  
AoB,qBAAO,O;YAAP,uB;WACpB,eA3EmB,QA2EJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YAC  
I,QAAQ,sBAAK,CAAL,C;YACR,QA9Ee,QA8EP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,U  
AAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QApFP,yB;O;KANJ,C;mFASA,yB;MAjhEI,8D;MAihEJ,sC;QAM  
W,sB;;UAuFP,IAAI,mBAAJ,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,sBAAK,CAAL,C;UACd,gBAhnEgB,cA  
AR,iBAAQ,C;UAinEhB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eA3FmB,QA2FJ,CAAS,O  
AAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,sBAAK,CAAL,C;YACR,QA9Fe,QA8FP,CAAS,CAA  
T,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QApGP,yB;O;  
KANJ,C;+FASA,yB;MALjEI,8D;MAkjEJ,sC;QASI,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,cAAc,sBAAK,CAAL,  
C;QACd,gBA7jEgB,cA6jEA,SA7jER,QAAQ,C;QA8jEhB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe  
,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,sBAAK,CAAL,C;UACR,QAAQ,SAAS,CAA  
T,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KAtBX,C;+F  
AyBA,yB;MAnkEI,8D;MAmkEJ,sC;QASI,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,cAAc,sBAAK,CAAL,C;QAC  
d,gBA9kEgB,cA8kEA,SA9kER,QAAQ,C;QA+kEhB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SA  
AS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,sBAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C  
;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KAtBX,C;+FAy  
BA,yB;MAPlEI,8D;MAolEI,sC;QASI,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,cAAc,sBAAK,CAAL,C;QACd,gB  
A/IEgB,cA+IEA,SA/IER,QAAQ,C;QAgmEhB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OA  
AT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,sBAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UAC

R,IAAI,2BAAW,CAAX,KA AJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KAtBX,C;+FAyBA,yB ;MArmEI,8D;MAqmeJ,sC;QASI,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,cAAc,sBAAK,CAAL,C;QACd,gBAhn EgB,cAgnEA,SAhnER,QAAQ,C;QAinEhB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT, C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,sBAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IA AI,2BAAW,CAAX,KA AJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KAtBX,C;kFAyBA,yB;MA AA,sE;MAtpEI,8D;MpBnwHJ,iB;MoBy5LA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eA Ae,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAtqEG,cAAR,iBAAQ,C;QAsqEhB,aAAU,CAAV,iB;UACI,QAAQ,S AAS,sBAAK,CAAL,CAAT,C;UACR,WpBn6LG,MAAO,KoBm6LO,QpBn6LP,EoBm6LiB,CpBn6LjB,C;;QoBq6 Ld,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MARqEI,8D;MpB3wHJ,iB;MoBg7LA,sC;QAgBiB,Q;QAFb,IA AI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OArrEG,cAAR,iBAAQ, C;QAqrEhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpB17LG,MAAO,KoB07LO ,QpB17LP,EoB07LiB,CpB17LjB,C;;QoB47Ld,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MAprEI,8D;MpB nxHJ,iB;MoBu8LA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL ,CAAT,C;QACF,OAprEG,cAAR,iBAAQ,C;QAosEhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CA AT,C;UACR,WpBj9LG,MAAO,KoBi9LO,QpBj9LP,EoBi9LiB,CpBj9LjB,C;;QoBm9Ld,OAAO,Q;O;KApBX,C;m FAuBA,yB;MAAA,sE;MAnsEI,8D;MpB3xHJ,iB;MoB89LA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM ,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAntEG,cAAR,iBAAQ,C;QAmtEhB,aAAU,CAAV,iB; UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBx+LG,MAAO,KoBw+LO,QpBx+LP,EoBw+LiB,CpBx +LjB,C;;QoB0+Ld,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MAIvEI,8D;MpB9wHJ,iB;MoBggMA,sC;QA gBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAlwEG, cAAR,iBAAQ,C;QakwEhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpB1gMG, MAAO,KoB0gMO,QpB1gMP,EoB0gMiB,CpB1gMjB,C;;QoB4gMd,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA, sE;MAjwEI,8D;MpBtxHJ,iB;MoBuhMA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe, SAAS,sBAAK,CAAL,CAAT,C;QACF,OAJxEG,cAAR,iBAAQ,C;QAixEhB,aAAU,CAAV,iB;UACI,QAAQ,SAA S,sBAAK,CAAL,CAAT,C;UACR,WpBjiMG,MAAO,KoBiiMO,QpBjiMP,EoBiiMiB,CpBjiMjB,C;;QoBmiMd,OA AO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MAhxEI,8D;MpB9xHJ,iB;MoB8iMA,sC;QAgBiB,Q;QAFb,IAAI,mB BAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAhYEG,cAAR,iBAAQ,C;QAgY EhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBxjMG,MAAO,KoBwjMO,QpBxj MP,EoBwjMiB,CpBxjMjB,C;;QoB0jMd,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MA/xEI,8D;MpBtyHJ,i B;MoBqkMA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAA T,C;QACF,OAYEG,cAAR,iBAAQ,C;QA+yEhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C; UACR,WpB/kMG,MAAO,KoB+kMO,QpB/kMP,EoB+kMiB,CpB/kMjB,C;;QoBilMd,OAAO,Q;O;KApBX,C;mF AuBA,yB;MAAA,sE;MA90EI,8D;MA80EJ,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe, SAAS,sBAAK,CAAL,CAAT,C;QACF,OA51EG,cAAR,iBAAQ,C;QA41EhB,aAAU,CAAV,iB;UACI,QAAQ,SAA S,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KA AJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KApBX ,C;mFAuBA,yB;MAAA,sE;MA71EI,8D;MA61EJ,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB, eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA32EG,cAAR,iBAAQ,C;QA22EhB,aAAU,CAAV,iB;UACI,QAA Q,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KA AJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O; KApBX,C;mFAuBA,yB;MAAA,sE;MA52EI,8D;MA42EJ,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B; QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA13EG,cAAR,iBAAQ,C;QA03EhB,aAAU,CAAV,iB;UA CI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KA AJ,C;YACI,WAAW,C;;QAGnB,OA AO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MA33EI,8D;MA23EJ,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,M AAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAz4EG,cAAR,iBAAQ,C;QAY4EhB,aAAU,CAA V,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KA AJ,C;YACI,WAAW,C;;QA GnB,OAAO,Q;O;KApBX,C;8FAuBA,yB;MA16EI,8D;MpBnwHJ,iB;MoB6qMA,sC;QACiB,Q;QAFb,IAAI,mBAA J,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAx7EG,cAAR,iBAAQ,C;QAw7EhB,a AAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBrrMG,MAAO,KoBqrMO,QpBrrMP,Eo BqrMiB,CpBrrMjB,C;;QoBurMd,OAAO,Q;O;KAlBX,C;+FAqBA,yB;MAv7EI,8D;MpB3wHJ,iB;MoBksMA,sC;Q

AciB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAr8EG,cA  
AR,iBAAQ,C;QAq8EhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpB1sMG,MA  
AO,KoB0sMO,QpB1sMP,EoB0sMiB,CpB1sMjB,C;;QoB4sMd,OAAO,Q;O;KAlBX,C;+FAqBA,yB;MAp8EI,8D;  
MpBnxHJ,iB;MoButMA,sC;QAcIB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAA  
L,CAAT,C;QACF,OAI9EG,cAAR,iBAAQ,C;QAk9EhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CA  
AT,C;UACR,WpB/tMG,MAAO,KoB+tMO,QpB/tMP,EoB+tMiB,CpB/tMjB,C;;QoBiuMd,OAAO,Q;O;KAlBX,C;  
+FAqBA,yB;MAj9EI,8D;MpB3xHJ,iB;MoB4uMA,sC;QAcIB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,e  
AAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA/9EG,cAAR,iBAAQ,C;QA+9EhB,aAAU,CAAV,iB;UACI,QAAQ,  
SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBpvMG,MAAO,KoBovMO,QpBpvMP,EoBovMiB,CpBpvMjB,C;;QoB  
svMd,OAAO,Q;O;KAlBX,C;+FAqBA,yB;MA9/EI,8D;MpB9wHJ,iB;MoB4wMA,sC;QAcIB,Q;QAFb,IAAI,mB  
AAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA5gFG,cAAR,iBAAQ,C;QA4gFhB,  
aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBpxMG,MAAO,KoBoxMO,QpBpxMP  
,EoBoxMiB,CpBpxMjB,C;;QoBsxMd,OAAO,Q;O;KAlBX,C;+FAqBA,yB;MA3gFI,8D;MpBtxHJ,iB;MoBiyMA,s  
C;QAcIB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAZhF  
G,cAAR,iBAAQ,C;QAYhFhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBzyMG,  
MAAO,KoByyMO,QpBzyMP,EoByyMiB,CpBzyMjB,C;;QoB2yMd,OAAO,Q;O;KAlBX,C;+FAqBA,yB;MAxhFI,  
8D;MpB9xHJ,iB;MoBszMA,sC;QAcIB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,C  
AAL,CAAT,C;QACF,OAtiFG,cAAR,iBAAQ,C;QAsiFhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,C  
AAT,C;UACR,WpB9zMG,MAAO,KoB8zMO,QpB9zMP,EoB8zMiB,CpB9zMjB,C;;QoBg0Md,OAAO,Q;O;KAlB  
X,C;+FAqBA,yB;MAriFI,8D;MpBtyHJ,iB;MoB20MA,sC;QAcIB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACt  
B,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAnjFG,cAAR,iBAAQ,C;QAmjFhB,aAAU,CAAV,iB;UACI,QA  
AQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBn1MG,MAAO,KoBm1MO,QpBn1MP,EoBm1MiB,CpBn1MjB,C;  
;QoBq1Md,OAAO,Q;O;KAlBX,C;+FAqBA,yB;MAIIFI,8D;MAkiFJ,sC;QAYiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,  
OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA9IFG,cAAR,iBAAQ,C;QA8IFhB,aAAU,CAAV,  
iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGn  
B,OAAO,Q;O;KAlBX,C;+FAqBA,yB;MAIFI,8D;MAIFI,sC;QAYiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;  
QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA3mFG,cAAR,iBAAQ,C;QA2mFhB,aAAU,CAAV,iB;UA  
CI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OA  
AO,Q;O;KAlBX,C;+FAqBA,yB;MA5mFI,8D;MA4mFJ,sC;QAYiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QA  
CtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAxnFG,cAAR,iBAAQ,C;QAwnFhB,aAAU,CAAV,iB;UACI,  
QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,  
Q;O;KAlBX,C;+FAqBA,yB;MAznFI,8D;MAynFJ,sC;QAYiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,e  
AAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAroFG,cAAR,iBAAQ,C;QAqoFhB,aAAU,CAAV,iB;UACI,QAAQ,  
SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;K  
AlBX,C;2FAqBA,yB;MAAA,sE;MATqFI,8D;MASqFJ,kD;QAcIB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QA  
CrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAprFG,cAAR,iBAAQ,C;QAorFhB,aAAU,CAAV,iB;UACI,Q  
AAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAaIB,CAAX,GAakC,CAAt  
C,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KApBX,C;0FAuBA,yB;MAAA,sE;MArrFI,8D;MAqrFJ,kD;QAcIB,Q;  
QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAnsFG,cAAR,i  
BAAQ,C;QAmsFhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,  
QAAR,EAakB,CAaIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KApBX,C;2FAuBA,yB  
;MAAA,sE;MApsFI,8D;MAosFJ,kD;QAcIB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sB  
AAK,CAAL,CAAT,C;QACF,OAltFG,cAAR,iBAAQ,C;QAktFhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,  
CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAaIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;  
;QAGnB,OAAO,Q;O;KApBX,C;2FAuBA,yB;MAAA,sE;MAntFI,8D;MAmtFJ,kD;QAcIB,Q;QAFb,IAAI,mBAAJ,  
C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAjuFG,cAAR,iBAAQ,C;QAiuFhB,a  
AAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAaI  
B,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KApBX,C;uGAuBA,yB;MAIwFI,8D;MAkwF



KANJ,C;mFASA,yB;MAhhGI,8D;MAghGJ,sC;QAMW,sB;;UAuEP,IAAI,mBAAJ,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,sBAAK,CAAL,C;UACd,gBA/lGgB,cAAR,iBAAQ,C;UAgmGhB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eA3EmB,QA2EJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,sBAAK,CAAL,C;YACR,QA9Ee,QA8EP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QApFP,yB;O;KANJ,C;mFASA,yB;MAjhGI,8D;MAihGJ,sC;QAMW,sB;;UAuFP,IAAI,mBAAJ,C;YAAe,qBAAO,I;YAAP,uB;WACf,cAAc,sBAAK,CAAL,C;UACd,gBAhnGgB,cAAR,iBAAQ,C;UAinGhB,IAAI,cAAa,CAAjB,C;YAAoB,qBAAO,O;YAAP,uB;WACpB,eA3FmB,QA2FJ,CAAS,OAAT,C;UACf,aAAU,CAAV,OAAa,SAAb,M;YACI,QAAQ,sBAAK,CAAL,C;YACR,QA9Fe,QA8FP,CAAS,CAAT,C;YACR,IAAI,2BAAW,CAAX,KAAJ,C;cACI,UAAU,C;cACV,WAAW,C;;UAGnB,qBAAO,O;;;QApGP,yB;O;KANJ,C;+FASA,yB;MALjGI,8D;MAkjGJ,sC;QASI,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,cAAc,sBAAK,CAAL,C;QACd,gBA7jGgB,cA6jGA,SA7jGR,QAAQ,C;QA8jGhB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,sBAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KATBX,C;+FAyBA,yB;MAnkGI,8D;MAmkGJ,sC;QASI,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,cAAc,sBAAK,CAAL,C;QACd,gBA9kGgB,cA8kGA,SA9kGR,QAAQ,C;QA+kGhB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,sBAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KATBX,C;+FAyBA,yB;MAplGI,8D;MAolGJ,sC;QASI,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,cAAc,sBAAK,CAAL,C;QACd,gBA/lGgB,cA+lGA,SA/lGR,QAAQ,C;QAgmGhB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,sBAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KATBX,C;+FAyBA,yB;MArmGI,8D;MAqmGJ,sC;QASI,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,cAAc,sBAAK,CAAL,C;QACd,gBAhnGgB,cAgnGA,SAhnGR,QAAQ,C;QAinGhB,IAAI,cAAa,CAAjB,C;UAAoB,OAAO,O;QAC3B,eAAe,SAAS,OAAT,C;QACf,aAAU,CAAV,OAAa,SAAb,M;UACI,QAAQ,sBAAK,CAAL,C;UACR,QAAQ,SAAS,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,UAAU,C;YACV,WAAW,C;;QAGnB,OAAO,O;O;KATBX,C;kFAyBA,yB;MAAA,sE;MATpGI,8D;MpB/iHJ,iB;MoBqsNA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAtqGG,cAAR,iBAAQ,C;QAsqGhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpB/sNG,MAAO,KoB+sNO,QpB/sNP,EoB+sNiB,CpB/sNjB,C;;QoBitNd,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MARqGI,8D;MpBvjHJ,iB;MoB4tNA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OArrGG,cAAR,iBAAQ,C;QAqrGhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBtuNG,MAAO,KoBsuNO,QpBtuNP,EoBsuNiB,CpBtuNjB,C;;QoBwuNd,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MAprGI,8D;MpB/jHJ,iB;MoBmvNA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OApGG,cAAR,iBAAQ,C;QAosGhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpB7vNG,MAAO,KoB6vNO,QpB7vNP,EoB6vNiB,CpB7vNjB,C;;QoB+vNd,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MAnsGI,8D;MpBvkHJ,iB;MoB0wNA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAntGG,cAAR,iBAAQ,C;QAmtGhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBpxNG,MAAO,KoBoxNO,QpBpxNP,EoBoxNiB,CpBpxNjB,C;;QoBsxDnd,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MALvGI,8D;MpB1jHJ,iB;MoB4yNA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAlwGG,cAAR,iBAAQ,C;QakwGhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBtzNG,MAAO,KoBszNO,QpBtzNP,EoBszNiB,CpBtzNjB,C;;QoBwzNd,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MAjwGI,8D;MpBlkHJ,iB;MoBm0NA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAJxGG,cAAR,iBAAQ,C;QAixGhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpB70NG,MAAO,KoB60NO,QpB70NP,EoB60NiB,CpB70NjB,C;;QoB+0Nd,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MAhxGI,8D;MpB1kHJ,iB;MoB01NA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAhYGG,cAAR,iBAAQ,C;QAgYgHb,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBp2NG,MAAO,KoBo2NO,QpBp

2NP,EoBo2NiB,CpBp2NjB,C;;QoBs2Nd,OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MA/xGI,8D;MpBllHJ,i  
B;MoBi3NA,sC;QAgBiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAA  
T,C;QACF,OA/yGG,cAAR,iBAAQ,C;QA+yGhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;  
UACR,WpB33NG,MAAO,KoB23NO,QpB33NP,EoB23NiB,CpB33NjB,C;;QoB63Nd,OAAO,Q;O;KApBX,C;mF  
AuBA,yB;MAAA,sE;MA90GI,8D;MA80GJ,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe  
,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA51GG,cAAR,iBAAQ,C;QA41GhB,aAAU,CAAV,iB;UACI,QAAQ,SA  
AS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KApB  
X,C;mFAuBA,yB;MAAA,sE;MA71GI,8D;MA61GJ,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACr  
B,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA32GG,cAAR,iBAAQ,C;QA22GhB,aAAU,CAAV,iB;UACI,QA  
AQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;  
O;KApBX,C;mFAuBA,yB;MAAA,sE;MA52GI,8D;MA42GJ,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,  
6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA13GG,cAAR,iBAAQ,C;QA03GhB,aAAU,CAAV,iB;  
UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,  
OAAO,Q;O;KApBX,C;mFAuBA,yB;MAAA,sE;MA33GI,8D;MA23GJ,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAA  
e,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAz4GG,cAAR,iBAAQ,C;QAY4GhB,aAAU,  
CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;  
;QAGnB,OAAO,Q;O;KApBX,C;8FAuBA,yB;MA16GI,8D;MpB/iHJ,iB;MoBy9NA,sC;QACiB,Q;QAFb,IAAI,mB  
AAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAx7GG,cAAR,iBAAQ,C;QAw7Gh  
B,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBj+NG,MAAO,KoBi+NO,QpBj+NP,  
EoBi+NiB,CpBj+NjB,C;;QoBm+Nd,OAAO,Q;O;KAIBX,C;+FAqBA,yB;MAv7GI,8D;MpBvjHJ,iB;MoB8+NA,sC  
;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAr8GG,  
cAAR,iBAAQ,C;QAq8GhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBt/NG,M  
AAO,KoBs/NO,QpBt/NP,EoBs/NiB,CpBt/NjB,C;;QoBw/Nd,OAAO,Q;O;KAIBX,C;+FAqBA,yB;MAp8GI,8D;Mp  
B/jHJ,iB;MoBmgOA,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,C  
AAT,C;QACF,OAI9GG,cAAR,iBAAQ,C;QAK9GhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT  
,C;UACR,WpB3gOG,MAAO,KoB2gOO,QpB3gOP,EoB2gOiB,CpB3gOjB,C;;QoB6gOd,OAAO,Q;O;KAIBX,C;+  
FAqBA,yB;MAj9GI,8D;MpBvkHJ,iB;MoBwhOA,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eA  
Ae,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA/9GG,cAAR,iBAAQ,C;QA+9GhB,aAAU,CAAV,iB;UACI,QAAQ,S  
AAS,sBAAK,CAAL,CAAT,C;UACR,WpBhiOG,MAAO,KoBgiOO,QpBhiOP,EoBgiOiB,CpBhiOjB,C;;QoBkiOd,  
OAAO,Q;O;KAIBX,C;+FAqBA,yB;MA9/GI,8D;MpB1jHJ,iB;MoBwjOA,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UA  
Ae,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA5gHG,cAAR,iBAAQ,C;QA4gHhB,aAAU,C  
AAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBhkOG,MAAO,KoBgkOO,QpBhkOP,EoBgkOi  
B,CpBhkOjB,C;;QoBkkOd,OAAO,Q;O;KAIBX,C;+FAqBA,yB;MA3gHI,8D;MpBlkHJ,iB;MoB6kOA,sC;QACiB,  
Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAzhHG,cAAR,i  
BAAQ,C;QAYhHhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,WpBrlOG,MAAO,Ko  
BqlOO,QpBrlOP,EoBqlOiB,CpBrlOjB,C;;QoBulOd,OAAO,Q;O;KAIBX,C;+FAqBA,yB;MAxhHI,8D;MpB1kHJ,i  
B;MoBkmOA,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;  
QACF,OAtiHG,cAAR,iBAAQ,C;QAsiHhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UAC  
R,WpB1mOG,MAAO,KoB0mOO,QpB1mOP,EoB0mOiB,CpB1mOjB,C;;QoB4mOd,OAAO,Q;O;KAIBX,C;+FAq  
BA,yB;MAriHI,8D;MpBllHJ,iB;MoBunOA,sC;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,S  
AAS,sBAAK,CAAL,CAAT,C;QACF,OAnjHG,cAAR,iBAAQ,C;QAmjHhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS  
,sBAAK,CAAL,CAAT,C;UACR,WpB/nOG,MAAO,KoB+nOO,QpB/nOP,EoB+nOiB,CpB/nOjB,C;;QoBioOd,OA  
AO,Q;O;KAIBX,C;+FAqBA,yB;MAllHI,8D;MAkiHJ,sC;QAYiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACt  
B,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA9IHG,cAAR,iBAAQ,C;QA8IHhB,aAAU,CAAV,iB;UACI,QA  
AQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;  
O;KAIBX,C;+FAqBA,yB;MA/IHI,8D;MA+IHJ,sC;QAYiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAA  
e,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA3mHG,cAAR,iBAAQ,C;QA2mHhB,aAAU,CAAV,iB;UACI,QAAQ,S  
AAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAI

BX,C;+FAqBA,yB;MA5mHI,8D;MA4mHJ,sC;QAYiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,S  
AAS,sBAAK,CAAL,CAAT,C;QACF,OAXnHG,cAAR,iBAAQ,C;QAwnHhB,aAAU,CAAV,iB;UACI,QAAQ,SAA  
S,sBAAK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAIBX,  
C;+FAqBA,yB;MAznHI,8D;MAynHJ,sC;QAYiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,s  
BAAK,CAAL,CAAT,C;QACF,OArHG,cAAR,iBAAQ,C;QAqoHhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBA  
AK,CAAL,CAAT,C;UACR,IAAI,2BAAW,CAAX,KAAJ,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAIBX,C;2F  
AqBA,yB;MAAA,sE;MATqHI,8D;MAsqHJ,kD;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe  
,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAprHG,cAAR,iBAAQ,C;QAorHhB,aAAU,CAAV,iB;UACI,QAAQ,SA  
AS,sBAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YAC  
I,WAAW,C;;QAGnB,OAAO,Q;O;KApBX,C;0FAuBA,yB;MAAA,sE;MArrHI,8D;MAqrHJ,kD;QACiB,Q;QAFb,I  
AAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAnsHG,cAAR,iBAAQ,  
C;QAmsHhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,  
EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KApBX,C;2FAuBA,yB;MAA  
A,sE;MApsHI,8D;MAosHJ,kD;QACiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,  
CAAL,CAAT,C;QACF,OAltHG,cAAR,iBAAQ,C;QAktHhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAA  
L,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QA  
GnB,OAAO,Q;O;KApBX,C;2FAuBA,yB;MAAA,sE;MAntHI,8D;MAmtHJ,kD;QACiB,Q;QAFb,IAAI,mBAAJ,C;  
UAAe,MAAM,6B;QACrB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OajuHG,cAAR,iBAAQ,C;QAiuHhB,aA  
AU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,  
CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KApBX,C;uGAuBA,yB;MALwHI,8D;MAkwHJ  
,kD;QAYiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OA9  
wHG,cAAR,iBAAQ,C;QA8wHhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,U  
AAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAIBX,C  
;sGAqBA,yB;MA/wHI,8D;MA+wHJ,kD;QAYiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,s  
BAAK,CAAL,CAAT,C;QACF,OA3xHG,cAAR,iBAAQ,C;QA2xHhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBA  
AK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAA  
W,C;;QAGnB,OAAO,Q;O;KAIBX,C;uGAqBA,yB;MA5xHI,8D;MA4xHJ,kD;QAYiB,Q;QAFb,IAAI,mBAAJ,C;U  
AAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OAxyHG,cAAR,iBAAQ,C;QAwyHhB,aAAU,  
CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,SAAQ,QAAR,EAakB,CAAIB,CA  
AX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAIBX,C;uGAqBA,yB;MAzyHI,8D;MAyyHJ,kD;  
QAYiB,Q;QAFb,IAAI,mBAAJ,C;UAAe,OAAO,I;QACtB,eAAe,SAAS,sBAAK,CAAL,CAAT,C;QACF,OArzHG,  
cAAR,iBAAQ,C;QAqzHhB,aAAU,CAAV,iB;UACI,QAAQ,SAAS,sBAAK,CAAL,CAAT,C;UACR,IAAI,UAAW,  
SAAQ,QAAR,EAakB,CAAIB,CAAX,GAakC,CAAtC,C;YACI,WAAW,C;;QAGnB,OAAO,Q;O;KAIBX,C;IAqB  
A,iC;MAQiB,Q;MAFb,IAAI,mBAAJ,C;QAae,OAAO,I;MActB,UAAU,sBAAK,CAAL,C;MACG,OA91HG,gBA  
AR,iBAAQ,C;MA81HhB,aAAU,CAAV,iB;QACI,QAAQ,sBAAK,CAAL,C;QACR,InC5mP8D,YmC4mP1D,GnC5  
mP2E,KAAjB,EmC4mPpD,CnC5mPiF,KAA7B,CmC4mP1D,IAAJ,C;UAAa,MAAM,C;;MAEvB,OAAO,G;K;IAG  
X,iC;MAQiB,Q;MAFb,IAAI,mBAAJ,C;QAae,OAAO,I;MActB,UAAU,sBAAK,CAAL,C;MACG,OAr2HG,gBA  
AR,iBAAQ,C;MAq2HhB,aAAU,CAAV,iB;QACI,QAAQ,sBAAK,CAAL,C;QACR,InBnnP+D,amBmnP3D,GnBnn  
P6E,KAAIB,EmBmnPrD,CnBnnPmF,KAA9B,CmBmnP3D,IAAJ,C;UAAa,MAAM,C;;MAEvB,OAAO,G;K;IAGX  
,iC;MAQiB,Q;MAFb,IAAI,mBAAJ,C;QAae,OAAO,I;MActB,UAAU,sBAAK,CAAL,C;MACG,OA52HG,gBAA  
R,iBAAQ,C;MA42HhB,aAAU,CAAV,iB;QACI,QAAQ,sBAAK,CAAL,C;QACR,IpC1pP4E,0BoC0pPxE,GpC/6O  
8B,KAAL,GAAiB,GA3O8B,EoC0pPIE,CpC/6OwB,KAAL,GAAiB,GA3O8B,CoC0pPxE,IAAJ,C;UAAa,MAAM,  
C;;MAEvB,OAAO,G;K;IAGX,iC;MAQiB,Q;MAFb,IAAI,mBAAJ,C;QAae,OAAO,I;MActB,UAAU,sBAAK,CA  
AL,C;MACG,OAn3HG,gBAAR,iBAAQ,C;Mam3HhB,aAAU,CAAV,iB;QACI,QAAQ,sBAAK,CAAL,C;QACR,I  
ICjqP6E,0BkCiqPzE,GIC77O8B,KAAL,GAAiB,KApO+B,EkCiqPnE,CIC77OwB,KAAL,GAAiB,KApO+B,CkCiq  
PzE,IAAJ,C;UAAa,MAAM,C;;MAEvB,OAAO,G;K;IAGX,2C;MAKI,OAAO,4BAAc,UAd,C;K;IAGX,2C;MAK  
I,OAAO,4BAAc,UAd,C;K;IAGX,2C;MAKI,OAAO,4BAAc,UAd,C;K;IAGX,2C;MAKI,OAAO,4BAAc,UAd,  
C;K;IAGX,iD;MAQiB,Q;MAFb,IAAI,mBAAJ,C;QAae,OAAO,I;MActB,UAAU,sBAAK,CAAL,C;MACG,OA17

HG,gBAAR,iBAAQ,C;MA07HhB,aAAU,CAAV,iB;QACI,QAAQ,sBAAK,CAAL,C;QACR,IAAI,UAAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IAGX,iD;MAQiB,Q;MAFb,IAAI,mBAAJ,C;QAAe,OAAO,I;MACtB,UAAU,sBAAK,CAAL,C;MACG,OAj8HG,gBAAR,iBAAQ,C;MAi8HhB,aAAU,CAAV,iB;QACI,QAAQ,sBAAK,CAAL,C;QACR,IAAI,UAAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IAGX,iD;MAQiB,Q;MAFb,IAAI,mBAAJ,C;QAAe,OAAO,I;MACtB,UAAU,sBAAK,CAAL,C;MACG,OAx8HG,gBAAR,iBAAQ,C;MAw8HhB,aAAU,CAAV,iB;QACI,QAAQ,sBAAK,CAAL,C;QACR,IAAI,UAAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;IAGX,iD;MAQiB,Q;MAFb,IAAI,mBAAJ,C;QAAe,OAAO,I;MACtB,UAAU,sBAAK,CAAL,C;MACG,OA/8HG,gBAAR,iBAAQ,C;MA+8HhB,aAAU,CAAV,iB;QACI,QAAQ,sBAAK,CAAL,C;QACR,IAAI,UAAW,SAAQ,GAAR,EAAa,CAAb,CAAX,GAA6B,CAAjC,C;UAAoC,MAAM,C;;MAE9C,OAAO,G;K;iFAGX,qB;MASI,OAAO,mB;K;iFAGX,qB;MASI,OAAO,mB;K;iFAGX,qB;MASI,OAAO,mB;K;iFAGX,qB;MASI,OAAO,mB;K;iFAGX,gC;MASoB,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,K;;MACrD,OAAO,I;K;iFAGX,gC;MASoB,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,K;;MACrD,OAAO,I;K;iFAGX,gC;MASoB,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,K;;MACrD,OAAO,I;K;iFAGX,gC;MASoB,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,IAAI,UAAU,OAAV,CAAJ,C;UAAwB,OAAO,K;;MACrD,OAAO,I;K;qFAGX,6B;MAOmC,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,OAAO,OAAP,C;;MAArC,gB;K;qFAGJ,6B;MAOmC,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,OAAO,OAAP,C;;MAArC,gB;K;qFAGJ,6B;MAOmC,Q;MAAA,2B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAM,OAAO,OAAP,C;;MAArC,gB;K;mGAGJ,6B;MArEiB,gB;MADb,YAAY,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QAAM,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;MAGsEnB,gB;K;mGAGJ,6B;MArEiB,gB;MADb,YAAY,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QAAM,QAAO,cAAP,EAAO,sBAAP,WAAgB,IAAhB,C;;MAGsEnB,gB;K;qFAGJ,yB;MAAA,4F;MA9qII,8D;MA8qIJ,uC;QAmBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,sBAAK,CAAL,C;QACD,OAjsID,cAAR,iBAAQ,C;QAisIhB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,sBAAK,KAAL,CAAvB,C;;QAEIB,OAAO,W;O;KAtBX,C;qFAyBA,yB;MAAA,4F;MA/rII,8D;MA+rII,uC;QAmBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,sBAAK,CAAL,C;QACD,OAltID,cAAR,iBAAQ,C;QAktIhB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,sBAAK,KAAL,CAAvB,C;;QAEIB,OAAO,W;O;KAtBX,C;qFAyBA,yB;MAAA,4F;MAjuII,8D;MAiuIJ,uC;QAmBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,sBAAK,CAAL,C;QACD,OApvID,cAAR,iBAAQ,C;QAovIhB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,sBAAK,KAAL,CAAvB,C;;QAEIB,OAAO,W;O;KAtBX,C;mGayBA,yB;MAAA,4F;MAIxII,8D;MAkxII,uC;QAmBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,sBAAK,CAAL,C;QACD,OArzID,cAAR,iBAAQ,C;QAszIhB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAAsV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAtBX,C;mGayBA,yB;MAAA,4F;MApzII,8D;MAozII,uC;QAmBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,sBAAK,CAAL,C;QACD,OAv0ID,cAAR,iBAAQ,C;QAu0IhB,iBAAc,CAAd,yB;UACI,cAAc,UAAU,KAAsV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAtBX,C;mGayBA,yB;MAAA,4F;MAR0II,8D;MAq0IJ,uC;QAmBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,MAAM,mCAA8B,+BAA9B,C;QACV,kBAAkB,sBAAK,CAAL,C;QACD,O

Ax1ID,cAAR,iBAAQ,C;QAw1IhB,iBAAC,CAAd,yB;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KAtBX,C;+GAYBA,yB;Mat3II,8D;MAS3IJ,uC;QAKBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,OAAO,I;QACX,kBAakB,sBAAK,CAAL,C;QACD,OAx4ID,cAAR,iBAAQ,C;QAw4IhB,iBAAC,CAAd,yB;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KArBX,C;+GAWBA,yB;Mat4II,8D;MAS4IJ,uC;QAKBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,OAAO,I;QACX,kBAakB,sBAAK,CAAL,C;QACD,OAx5ID,cAAR,iBAAQ,C;QAw5IhB,iBAAC,CAAd,yB;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KArBX,C;+GAWBA,yB;Mat5II,8D;MAS5IJ,uC;QAKBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,OAAO,I;QACX,kBAakB,sBAAK,CAAL,C;QACD,OAx6ID,cAAR,iBAAQ,C;QAw6IhB,iBAAC,CAAd,yB;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KArBX,C;+GAWBA,yB;Mat6II,8D;MAS6IJ,uC;QAKBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,OAAO,I;QACX,kBAakB,sBAAK,CAAL,C;QACD,OAx7ID,cAAR,iBAAQ,C;QAw7IhB,iBAAC,CAAd,yB;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,CAA9B,C;;QAEIB,OAAO,W;O;KArBX,C;+GAWBA,yB;Mat9II,8D;MAS9IJ,uC;QAmBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,OAAO,I;QACX,kBAakB,sBAAK,CAAL,C;QACD,OAz+ID,cAAR,iBAAQ,C;QAY+IhB,iBAAC,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,sBAAK,KAAL,CAAvB,C;;QAEIB,OAAO,W;O;KAtBX,C;+GAYBA,yB;MAv+II,8D;MAu+IJ,uC;QAmBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,OAAO,I;QACX,kBAakB,sBAAK,CAAL,C;QACD,OAx1ID,cAAR,iBAAQ,C;QAO/IhB,iBAAC,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,sBAAK,KAAL,CAAvB,C;;QAEIB,OAAO,W;O;KAtBX,C;+GAYBA,yB;MAx/II,8D;MAw/IJ,uC;QAmBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,OAAO,I;QACX,kBAakB,sBAAK,CAAL,C;QACD,OAx3gJD,cAAR,iBAAQ,C;QA2gJhB,iBAAC,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,sBAAK,KAAL,CAAvB,C;;QAEIB,OAAO,W;O;KAtBX,C;+GAYBA,yB;MAzgjI,8D;MAyggJJ,uC;QAmBqB,Q;QAHjB,IAAI,mBAAJ,C;UACI,OAAO,I;QACX,kBAakB,sBAAK,CAAL,C;QACD,OAx5hJD,cAAR,iBAAQ,C;QA4hJhB,iBAAC,CAAd,yB;UACI,cAAc,UAAU,WAAV,EAAuB,sBAAK,KAAL,CAAvB,C;;QAEIB,OAAO,W;O;KAtBX,C;+FAYBA,yB;MAAA,4F;MA1jJI,8D;MA0jJJ,uC;QAKB0B,UAEU,M;QAJhC,YA1kJgB,cAAR,iBAAQ,C;QA2kJhB,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,uBAAI,YAAJ,EAAL,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,uBAAI,cAAJ,EAAL,sBAAJ,UAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KAtBX,C;+FAYBA,yB;MAAA,4F;MA3kJI,8D;MA2kJJ,uC;QAKB0B,UAEU,M;QAJhC,YA3lJgB,cAAR,iBAAQ,C;QA4lJhB,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,uBAAI,YAAJ,EAAL,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,uBAAI,cAAJ,EAAL,sBAAJ,UAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KAtBX,C;+FAYBA,yB;MAAA,4F;MA5lJI,8D;MA4lJJ,uC;QAKB0B,UAEU,M;QAJhC,YA5mJgB,cAAR,iBAAQ,C;QA6mJhB,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,uBAAI,YAAJ,EAAL,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,uBAAI,cAAJ,EAAL,sBAAJ,UAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KAtBX,C;+FAYBA,yB;MAAA,4F;MA7mJI,8D;MA6mJJ,uC;QAKB0B,UAEU,M;QAJhC,YA7nJgB,cAAR,iBAAQ,C;QA8nJhB,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,uBAAI,YAAJ,EAAL,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,uBAAI,cAAJ,EAAL,sBAAJ,UAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KAtBX,C;+FAYBA,yB;MAAA,4F;MA9pJI,8D;MA8pJJ,uC;QAKB0B,Q;QAFtB,YA9qJgB,cAAR,iBAAQ,C;QA+qJhB,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,uBAAI,YAAJ,EAAL,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,sBAAL,KAAL,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAvBX,C;+6GA0BA,yB;MAAA,4F;MAhrJI,8D;MAgrJJ,uC;QAKB0B,Q;QAFtB,YAhsJgB,cAAR,iBAAQ,C;QAisJhB,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,uBAAI,YAAJ,EAAL,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,sBAAL,KAAL,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAvBX,C;+6GA0BA,yB;MAAA,4F;MAIsJI,8D;MAksJJ,uC;QAKB0B,Q;QAFtB,YAltJgB,cAAR,iBAAQ,C;QAMtJhB,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,uBAAI,YAAJ,EAAL,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,sBAAL,KAAL,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,W;O;KAvBX,C;+6GA0BA,yB;MAAA,4F;MAptJI,8D;MAotJJ,uC;QAKB0B,Q;QAFtB,YApuJgB,cAAR,iBAAQ,C;QAquJhB,IAAI,QAAQ,CAAZ,C;UAAe,MAAM,mCAA8B,+BAA9B,C;QACrB,kBAakB,uBAAI,YAAJ,EAAL,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,sBAAL,KAAL,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,O

AAO,W;O;KAvBX,C;yHA0BA,yB;MAtwJI,8D;MAswJJ,uC;QAIb0B,Q;QAFtB,YArxJgB,cAAR,iBAAQ,C;QAsx  
JhB,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,uBAAI,YAAJ,EAAI,oBAAJ,Q;QACIB,OAAO,SAAS,  
CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,sBAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OA  
AO,W;O;KAtBX,C;yHAyBA,yB;MAvxJI,8D;MAuxJJ,uC;QAIb0B,Q;QAFtB,YAtyJgB,cAAR,iBAAQ,C;QAuyJh  
B,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,uBAAI,YAAJ,EAAI,oBAAJ,Q;QACIB,OAAO,SAAS,  
CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,sBAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAA  
O,W;O;KAtBX,C;yHAyBA,yB;MAxyJI,8D;MAwyJJ,uC;QAIb0B,Q;QAFtB,YAvzJgB,cAAR,iBAAQ,C;QAwzJh  
B,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,uBAAI,YAAJ,EAAI,oBAAJ,Q;QACIB,OAAO,SAAS,  
CAAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,sBAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAA  
O,W;O;KAtBX,C;yHAyBA,yB;MAzzJI,8D;MAyzJJ,uC;QAIb0B,Q;QAFtB,YAx0JgB,cAAR,iBAAQ,C;QAY0JhB,  
IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,uBAAI,YAAJ,EAAI,oBAAJ,Q;QACIB,OAAO,SAAS,C  
AAhB,C;UACI,cAAc,UAAU,KAAV,EAAiB,sBAAI,KAAJ,CAAjB,EAA6B,WAA7B,C;UACd,qB;;QAEJ,OAAO,  
W;O;KAtBX,C;2GAYBA,yB;MA12JI,8D;MA02JJ,uC;QAKb0B,UAEU,M;QAJhC,YA13JgB,cAAR,iBAAQ,C;QA  
23JhB,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,uBAAI,YAAJ,EAAI,oBAAJ,Q;QACIB,OAAO,SA  
AS,CAAhB,C;UACI,cAAc,UAAU,uBAAI,cAAJ,EAAI,sBAAJ,UAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;  
KAtBX,C;2GAYBA,yB;MA33JI,8D;MA23JJ,uC;QAKb0B,UAEU,M;QAJhC,YA34JgB,cAAR,iBAAQ,C;QA44Jh  
B,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,uBAAI,YAAJ,EAAI,oBAAJ,Q;QACIB,OAAO,SAAS,  
CAAhB,C;UACI,cAAc,UAAU,uBAAI,cAAJ,EAAI,sBAAJ,UAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KAt  
BX,C;2GAYBA,yB;MA54JI,8D;MA44JJ,uC;QAKb0B,UAEU,M;QAJhC,YA55JgB,cAAR,iBAAQ,C;QA65JhB,IA  
AI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,uBAAI,YAAJ,EAAI,oBAAJ,Q;QACIB,OAAO,SAAS,CAA  
hB,C;UACI,cAAc,UAAU,uBAAI,cAAJ,EAAI,sBAAJ,UAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KAtBX,C  
;2GAYBA,yB;MA75JI,8D;MA65JJ,uC;QAKb0B,UAEU,M;QAJhC,YA76JgB,cAAR,iBAAQ,C;QA86JhB,IAAI,Q  
AAQ,CAAZ,C;UAAe,OAAO,I;QACtB,kBAakB,uBAAI,YAAJ,EAAI,oBAAJ,Q;QACIB,OAAO,SAAS,CAAhB,C  
;UACI,cAAc,UAAU,uBAAI,cAAJ,EAAI,sBAAJ,UAAV,EAAwB,WAAxB,C;;QAEIB,OAAO,W;O;KAtBX,C;+F  
AyBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAKBoB,Q;QAHhB,IAAI,mBAAJ,C;UAAe,OAAO,OAAO,OAAP,  
C;QACc,kBAAvB,eAAa,iBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,arBrRO,W;QqBstRP,kBAakB,O;QACF,2  
B;QAAhB,OAAGB,cAAhB,C;UAGB,yB;UACZ,cAAc,UAAU,WAAV,EAAuB,OAavB,C;UACd,MAAO,WAAI,  
WAAJ,C;;QAEX,OAAO,M;O;KAtBX,C;+FAyBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAKBoB,Q;QAHhB,IA  
AI,mBAAJ,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,iBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,  
arB9uRO,W;QqB+uRP,kBAakB,O;QACF,2B;QAAhB,OAAGB,cAAhB,C;UAGB,yB;UACZ,cAAc,UAAU,WAA  
V,EAAuB,OAavB,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAtBX,C;+FAyBA,yB;MAAA,gD;M  
AAA,gE;MAAA,gD;QAKBoB,Q;QAHhB,IAAI,mBAAJ,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,  
iBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,arBvwRO,W;QqBwwRP,kBAakB,O;QACF,2B;QAAhB,OAAGB,cA  
AhB,C;UAGB,yB;UACZ,cAAc,UAAU,WAAV,EAAuB,OAavB,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAA  
O,M;O;KAtBX,C;+FAyBA,yB;MAAA,gD;MAAA,gE;MAAA,gD;QAKBoB,Q;QAHhB,IAAI,mBAAJ,C;UAAe,O  
AAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,iBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,arBhyRO,W;QqBiyRP,  
kBAakB,O;QACF,2B;QAAhB,OAAGB,cAAhB,C;UAGB,yB;UACZ,cAAc,UAAU,WAAV,EAAuB,OAavB,C;U  
ACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAtBX,C;6GAYBA,yB;MAAA,gD;MAAA,gE;MAIIKI,0D;M  
AklKJ,gD;QAmBkB,gC;QAHd,IAAI,mBAAJ,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,iBAAO,C  
AAP,IAAb,C;QAA+B,8B;QAA5C,arB1zRO,W;QqB2zRP,kBAakB,O;QACJ,OArmKE,YAAR,iBAAQ,C;QAqmK  
F,mB;QAAA,kB;QAAA,kB;QAAAd,0D;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,CAA  
9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAvBX,C;6GA0BA,yB;MAAA,gD;MAAA,gE;MApm  
KI,0D;MAomKJ,gD;QAmBkB,gC;QAHd,IAAI,mBAAJ,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAvB,eAAa,  
iBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,arBp1RO,W;QqBq1RP,kBAakB,O;QACJ,OAvnKE,YAAR,iBAAQ,  
C;QAunKF,mB;QAAA,kB;QAAA,kB;QAAAd,0D;UACI,cAAc,UAAU,KAAV,EAAiB,WAAjB,EAA8B,sBAAK,K  
AAL,CAA9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAvBX,C;6GA0BA,yB;MAAA,gD;MAAA,  
gE;MATnKI,0D;MASnKJ,gD;QAmBkB,gC;QAHd,IAAI,mBAAJ,C;UAAe,OAAO,OAAO,OAAP,C;QACc,kBAAv  
B,eAAa,iBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,arB92RO,W;QqB+2RP,kBAakB,O;QACJ,OazoKE,YAAR,i

BAAQ,C;QAyoKF,mB;QAAA,kB;QAAA,kB;QAAd,0D;UACI,cAAc,UAAU,KAaV,EAAiB,WAAjB,EAA8B,sB  
AAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAvBX,C;6GA0BA,yB;MAAA,gD;  
MAAA,gE;MAxoKI,0D;MAwoKJ,gD;QAmBkB,gC;QAHd,IAAI,mBAAJ,C;UAAe,OAAO,OAAO,OAAP,C;QAC  
c,kBAAvB,eAAa,iBAAO,CAAP,IAAb,C;QAA+B,8B;QAA5C,arBx4RO,W;QqBy4RP,kBAaKB,O;QACJ,OA3pK  
E,YAAR,iBAAQ,C;QA2pKF,mB;QAAA,kB;QAAA,kB;QAAd,0D;UACI,cAAc,UAAU,KAaV,EAAiB,WAAjB,E  
AA8B,sBAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,WAAJ,C;;QAEX,OAAO,M;O;KAvBX,C;mGA0BA,yB;M  
AAA,qD;MAAA,gE;MAAA,uC;QakB0B,Q;QAHtB,IAAI,mBAAJ,C;UAAe,OAAO,W;QACtB,sBAaKB,sBAAK,  
CAAL,CAAIB,C;QACmC,kBAAtB,eAAgB,cAAhB,C;QAA8B,sBAAI,aAAJ,C;QAA3C,arBl6RO,W;QqBm6Re,q  
B;QAAtB,iBAAc,CAAd,wB;UACI,gBAAc,UAAU,aAAV,EAAuB,sBAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,  
aAAJ,C;;QAEX,OAAO,M;O;KATBX,C;mGAyBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QakB0B,Q;QAHtB,I  
AAI,mBAAJ,C;UAAe,OAAO,W;QACtB,sBAaKB,sBAAK,CAAL,CAAIB,C;QACoC,kBAAvB,eAAiB,cAAjB,C;  
QAA+B,sBAAI,aAAJ,C;QAA5C,arB37RO,W;QqB47Re,qB;QAAtB,iBAAc,CAAd,wB;UACI,gBAAc,UAAU,aAA  
V,EAAuB,sBAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KATBX,C;mGAyBA,yB;  
MAAA,qD;MAAA,gE;MAAA,uC;QakB0B,Q;QAHtB,IAAI,mBAAJ,C;UAAe,OAAO,W;QACtB,sBAaKB,sBAA  
K,CAAL,CAAIB,C;QACoC,kBAAvB,eAAiB,cAAjB,C;QAA+B,sBAAI,aAAJ,C;QAA5C,arBp9RO,W;QqBq9Re,q  
B;QAAtB,iBAAc,CAAd,wB;UACI,gBAAc,UAAU,aAAV,EAAuB,sBAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,  
aAAJ,C;;QAEX,OAAO,M;O;KATBX,C;mGAyBA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QakB0B,Q;QAHtB,I  
AAI,mBAAJ,C;UAAe,OAAO,W;QACtB,sBAaKB,sBAAK,CAAL,CAAIB,C;QACqC,kBAAxB,eAAkB,cAAIB,C;  
QAAgC,sBAAI,aAAJ,C;QAA7C,arB7+RO,W;QqB8+Re,qB;QAAtB,iBAAc,CAAd,wB;UACI,gBAAc,UAAU,aAA  
V,EAAuB,sBAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KATBX,C;iHAyBA,yB  
;MAAA,qD;MAAA,gE;MAAA,uC;QAmB0B,Q;QAHtB,IAAI,mBAAJ,C;UAAe,OAAO,W;QACtB,sBAaKB,sBA  
AK,CAAL,CAAIB,C;QACmC,kBAAtB,eAAgB,cAAhB,C;QAA8B,sBAAI,aAAJ,C;QAA3C,arBvgSO,W;QqBwgS  
e,qB;QAAtB,iBAAc,CAAd,wB;UACI,gBAAc,UAAU,KAaV,EAAiB,aAAjB,EAA8B,sBAAK,KAAL,CAA9B,C;  
UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAvBX,C;iHA0BA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QA  
mB0B,Q;QAHtB,IAAI,mBAAJ,C;UAAe,OAAO,W;QACtB,sBAaKB,sBAAK,CAAL,CAAIB,C;QACoC,kBAAvB,  
eAAiB,cAAjB,C;QAA+B,sBAAI,aAAJ,C;QAA5C,arBjiSO,W;QqBkiSe,qB;QAAtB,iBAAc,CAAd,wB;UACI,gBA  
Ac,UAAU,KAaV,EAAiB,aAAjB,EAA8B,sBAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAA  
O,M;O;KAvBX,C;iHA0BA,yB;MAAA,qD;MAAA,gE;MAAA,uC;QAmB0B,Q;QAHtB,IAAI,mBAAJ,C;UAAe,O  
AAO,W;QACtB,sBAaKB,sBAAK,CAAL,CAAIB,C;QACoC,kBAAvB,eAAiB,cAAjB,C;QAA+B,sBAAI,aAAJ,C;  
QAA5C,arB3jSO,W;QqB4jSe,qB;QAAtB,iBAAc,CAAd,wB;UACI,gBAAc,UAAU,KAaV,EAAiB,aAAjB,EAA8B  
,sBAAK,KAAL,CAA9B,C;UACd,MAAO,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAvBX,C;iHA0BA,yB;MAAA,q  
D;MAAA,gE;MAAA,uC;QAmB0B,Q;QAHtB,IAAI,mBAAJ,C;UAAe,OAAO,W;QACtB,sBAaKB,sBAAK,CAAL  
,CAAIB,C;QACqC,kBAAxB,eAAkB,cAAIB,C;QAAgC,sBAAI,aAAJ,C;QAA7C,arBrlSO,W;QqBslSe,qB;QAAtB,  
iBAAc,CAAd,wB;UACI,gBAAc,UAAU,KAaV,EAAiB,aAAjB,EAA8B,sBAAK,KAAL,CAA9B,C;UACd,MAAO  
,WAAI,aAAJ,C;;QAEX,OAAO,M;O;KAvBX,C;iFA0BA,yB;MAxZA,gD;MAAA,gE;MAwZA,gD;QAgBW,sB;;U  
AtZS,Q;UAHhB,IAAI,mBAAJ,C;YAAe,qBAAO,OAYZH,OAzZG,C;YAAP,uB;WACqB,kBAAvB,eAAa,iBAAO,  
CAAP,IAAb,C;UAA+B,sBAwZzB,OAxZyB,C;UAA5C,arBrTRO,W;UqBstRP,kBAuZmB,O;UAtZH,2B;UAAhB,O  
AAgB,cAAhB,C;YAAgB,yB;YACZ,cAqZwB,SArZV,CAAU,WAAV,EAAuB,OAAvB,C;YACd,MAAO,WAAI,W  
AAJ,C;;UAEX,qBAAO,M;;;QAKZP,yB;O;KAhBJ,C;iFamBA,yB;MAIZA,gD;MAAA,gE;MAkZA,gD;QAgBW,sB  
;;UAhZS,Q;UAHhB,IAAI,mBAAJ,C;YAAe,qBAAO,OAmZH,OAnZG,C;YAAP,uB;WACqB,kBAAvB,eAAa,iBA  
AO,CAAP,IAAb,C;UAA+B,sBAkZzB,OAIzyB,C;UAA5C,arB9uRO,W;UqB+uRP,kBAiZmB,O;UAhZH,2B;UAA  
hB,OAAgB,cAAhB,C;YAAgB,yB;YACZ,cAYwB,SAyYV,CAAU,WAAV,EAAuB,OAAvB,C;YACd,MAAO,WAAI,  
WAAJ,C;;UAEX,qBAAO,M;;;QA4YP,yB;O;KAhBJ,C;iFamBA,yB;MA5YA,gD;MAAA,gE;MA4YA,gD;Q  
AgBW,sB;;UA1YS,Q;UAHhB,IAAI,mBAAJ,C;YAAe,qBAAO,OA6YH,OA7YG,C;YAAP,uB;WACqB,kBAAvB,  
eAAa,iBAAO,CAAP,IAAb,C;UAA+B,sBA4YzB,OA5YyB,C;UAA5C,arBvwRO,W;UqBwwRP,kBA2YmB,O;UA  
1YH,2B;UAAhB,OAAgB,cAAhB,C;YAAgB,yB;YACZ,cAYwB,SAzYV,CAAU,WAAV,EAAuB,OAAvB,C;YA  
Cd,MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;;QAsYP,yB;O;KAhBJ,C;iFamBA,yB;MATYA,gD;MAAA,gE;M  
AsYA,gD;QAgBW,sB;;UApYS,Q;UAHhB,IAAI,mBAAJ,C;YAAe,qBAAO,OAuYH,OAyYG,C;YAAP,uB;WACq

B,kBAAvB,eAAa,iBAAO,CAAP,IAAb,C;UAA+B,sBA sYzB,OAtYyB,C;UAA5C,arBhyRO,W;UqBiyRP,kBAqY  
mB,O;UApYH,2B;UAAhB,OAAgB,cAAhB,C;YAAgB,yB;YACZ,cAmYwB,SAnYV,CAAU,WAAV,EAAuB,OA  
AvB,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,qBAAO,M;;;QAgYP,yB;O;KAhBJ,C;+FAmBA,yB;MAhYA,gD;  
MAAA,gE;MAIki,0D;MAK9KJ,gD;QAIbW,6B;;UA9XO,gC;UAHd,IAAI,mBAAJ,C;YAAe,4BAAO,OAIYI,OAj  
YJ,C;YAAP,8B;WACqB,kBAAvB,eAAa,iBAAO,CAAP,IAAb,C;UAA+B,sBAgYIB,OAhYkB,C;UAA5C,arB1zR  
O,W;UqB2zRP,kBA+X0B,O;UA9XZ,OArmKE,YAAR,iBAAQ,C;UAqmKF,mB;UAAA,kB;UAAA,kB;UAAAd,0D;  
YACI,cA6X+B,SA7XjB,CAAU,KAAV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,CAA9B,C;YACd,MAAO,WAAI  
,WAAJ,C;;UAEX,4BAAO,M;;;QA0XP,gC;O;KAjBJ,C;+FAoBA,yB;MA1XA,gD;MAAA,gE;MApmKI,0D;MA89  
KJ,gD;QAIbW,6B;;UAxXO,gC;UAHd,IAAI,mBAAJ,C;YAAe,4BAAO,OA2XI,OA3XJ,C;YAAP,8B;WACqB,kB  
AAvB,eAAa,iBAAO,CAAP,IAAb,C;UAA+B,sBA0XIB,OA1XkB,C;UAA5C,arBp1RO,W;UqBq1RP,kBAyX0B,O  
;UAxXZ,OAvnKE,YAAR,iBAAQ,C;UAunKF,mB;UAAA,kB;UAAA,kB;UAAAd,0D;YACI,cAuX+B,SAvXjB,CA  
AU,KAAV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,CAA9B,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,  
M;;;QAoXP,gC;O;KAjBJ,C;+FAoBA,yB;MApXA,gD;MAAA,gE;MAtnKI,0D;MA0+KJ,gD;QAIbW,6B;;UAIXO,  
gC;UAHd,IAAI,mBAAJ,C;YAAe,4BAAO,OAqXI,OA rXJ,C;YAAP,8B;WACqB,kBAAvB,eAAa,iBAAO,CAAP,I  
AAb,C;UAA+B,sBAoXIB,OApxkB,C;UAA5C,arB92RO,W;UqB+2RP,kBAmX0B,O;UAIXZ,OAzoKE,YAAR,iB  
AAQ,C;UAyoKF,mB;UAAA,kB;UAAA,kB;UAAAd,0D;YACI,cAiX+B,SAjXjB,CAAU,KAAV,EAAiB,WAAjB,E  
AA8B,sBAAK,KAAL,CAA9B,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,M;;;QA8WP,gC;O;KAjBJ,C;+F  
AoBA,yB;MA9WA,gD;MAAA,gE;MAxoKI,0D;MA s/KJ,gD;QAIbW,6B;;UA5WO,gC;UAHd,IAAI,mBAAJ,C;Y  
AAe,4BAAO,OA+WI,OA/WJ,C;YAAP,8B;WACqB,kBAAvB,eAAa,iBAAO,CAAP,IAAb,C;UAA+B,sBA8WIB,  
OA9WkB,C;UAA5C,arBx4RO,W;UqBy4RP,kBA6W0B,O;UA5WZ,OA3pKE,YAAR,iBAAQ,C;UA2pKF,mB;UA  
AA,kB;UAAA,kB;UAAAd,0D;YACI,cA2W+B,SA3WjB,CAAU,KAAV,EAAiB,WAAjB,EAA8B,sBAAK,KAAL,C  
AA9B,C;YACd,MAAO,WAAI,WAAJ,C;;UAEX,4BAAO,M;;;QA wWP,gC;O;KAjBJ,C;mFAoBA,yB;MAAA,wB;  
MAAA,sC;QAUoB,Q;QADhB,UAAgB,W;QACA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,MnCvWsiD,  
SmCuWsjD,GnCvW s2D,KAAK,GmCuWszD,SAAS,OAAT,CnCvWsoE,KAAX,IAAf,C;;QmCysrD,OAAO,G;O;  
KAbX,C;mFAGBA,yB;MAAA,wB;MAAA,sC;QAUoB,Q;QADhB,UAAgB,W;QACA,2B;QAAhB,OAAgB,cAAh  
B,C;UAAgB,yB;UACZ,MnCvXsiD,SmCuxSjD,GnCvXs2D,KAAK,GmCuxszD,SAAS,OAAT,CnCvXsoE,KAAX,I  
AAf,C;;QmCyXsrD,OAAO,G;O;KAbX,C;mFAGBA,yB;MAAA,wB;MAAA,sC;QAUoB,Q;QADhB,UAAgB,W;Q  
ACA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,MnCvySiD,SmCuySjD,GnCvyS2D,KAAK,GmCuySzD,S  
AAS,OAAT,CnCvySoE,KAAX,IAAf,C;;QmCysrD,OAAO,G;O;KAbX,C;mFAGBA,yB;MAAA,wB;MAAA,sC;Q  
AUoB,Q;QADhB,UAAgB,W;QACA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,MnCvzSiD,SmCuzSjD,G  
nCvzS2D,KAAK,GmCuzszD,SAAS,OAAT,CnCvzSoE,KAAX,IAAf,C;;QmCyzsrD,OAAO,G;O;KAbX,C;8FAGB  
A,+B;MAUoB,Q;MADhB,UAAkB,G;MACF,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,OAAO,SAAS,O  
AAT,C;;MAEX,OAAO,G;K;+FAGX,+B;MAUoB,Q;MADhB,UAAkB,G;MACF,2B;MAAhB,OAAgB,cAAhB,C;  
QAAgB,yB;QACZ,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;+FAGX,+B;MAUoB,Q;MADhB,UAAkB,G;MA  
CF,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;+FAGX,+B;  
MAUoB,Q;MADhB,UAAkB,G;MACF,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,OAAO,SAAS,OAAT,  
C;;MAEX,OAAO,G;K;kFAGX,+B;MAYoB,Q;MADhB,UAAoB,C;MACJ,2B;MAAhB,OAAgB,cAAhB,C;QAAg  
B,yB;QACZ,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;mFAGX,+B;MAYoB,Q;MADhB,UAAoB,C;MACJ,2B;  
MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;mFAGX,+B;MAYo  
B,Q;MADhB,UAAoB,C;MACJ,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,OAAO,SAAS,OAAT,C;;MAE  
X,OAAO,G;K;mFAGX,+B;MAYoB,Q;MADhB,UAAoB,C;MACJ,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;Q  
ACZ,OAAO,SAAS,OAAT,C;;MAEX,OAAO,G;K;mFAGX,+B;MAYoB,Q;MADhB,UAAe,C;MACC,2B;MAAhB,  
OAAgB,cAAhB,C;QAAgB,yB;QACZ,YAAO,SAAS,OAAT,CAAP,I;;MAEJ,OAAO,G;K;mFAGX,+B;MAYoB,Q;  
MADhB,UAAe,C;MACC,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,YAAO,SAAS,OAAT,CAAP,I;;MAE  
J,OAAO,G;K;mFAGX,+B;MAYoB,Q;MADhB,UAAe,C;MACC,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QAC  
Z,YAAO,SAAS,OAAT,CAAP,I;;MAEJ,OAAO,G;K;mFAGX,+B;MAYoB,Q;MADhB,UAAe,C;MACC,2B;MAAh  
B,OAAgB,cAAhB,C;QAAgB,yB;QACZ,YAAO,SAAS,OAAT,CAAP,I;;MAEJ,OAAO,G;K;mFAGX,yB;MAAA,S  
AWoB,gB;MAXpB,sC;QAYoB,Q;QADhB,Y;QACgB,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,cAAO,S

AAS,OAAT,CAAP,C;;QAEJ,OAAO,G;O;KafX,C;mFAkBA,yB;MAAA,SAWoB,gB;MAXpB,sC;QAYoB,Q;QA  
DhB,Y;QACgB,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,cAAO,SAAS,OAAT,CAAP,C;;QAEJ,OAAO,  
G;O;KafX,C;mFAkBA,yB;MAAA,SAWoB,gB;MAXpB,sC;QAYoB,Q;QADhB,Y;QACgB,2B;QAAhB,OAAgB,c  
AAhB,C;UAAgB,yB;UACZ,cAAO,SAAS,OAAT,CAAP,C;;QAEJ,OAAO,G;O;KafX,C;mFAkBA,yB;MAAA,SA  
WoB,gB;MAXpB,sC;QAYoB,Q;QADhB,Y;QACgB,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,cAAO,SA  
AS,OAAT,CAAP,C;;QAEJ,OAAO,G;O;KafX,C;mFAkBA,yB;MnC5xSA,6B;MmC4xSA,sC;QAaoB,Q;QADhB,U  
nC9xSmC,cmC8xSnB,CnC9xSmB,C;QmC+xSnB,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,MnClmTiD,c  
mCkmTjD,GnClmT2D,KAAK,GmCkmTzD,SAAS,OAAT,CnCImToE,KAAK,IAAf,C;;QmComTrD,OAAO,G;O;  
KAhBX,C;mFAmBA,yB;MnC/ySA,6B;MmC+ySA,sC;QAaoB,Q;QADhB,UnCjzSmC,cmCizSnB,CnCjzSmB,C;Q  
mCkzSnB,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,MnCrnTiD,cmCqnTjD,GnCrnT2D,KAAK,GmCqnT  
zD,SAAS,OAAT,CnCrnToE,KAAK,IAAf,C;;QmCunTrD,OAAO,G;O;KAhBX,C;mFAmBA,yB;MnCl0SA,6B;Mm  
Ck0SA,sC;QAaoB,Q;QADhB,UnCp0SmC,cmCo0SnB,CnCp0SmB,C;QmCq0SnB,2B;QAAhB,OAAgB,cAAhB,C;  
UAAgB,yB;UACZ,MnCxoTiD,cmCwoTjD,GnCxoT2D,KAAK,GmCwoTzD,SAAS,OAAT,CnCxoToE,KAAK,IA  
Af,C;;QmC0oTrD,OAAO,G;O;KAhBX,C;mFAmBA,yB;MnCr1SA,6B;MmCq1SA,sC;QAaoB,Q;QADhB,UnCv1S  
mC,cmCu1SnB,CnCv1SmB,C;QmCw1SnB,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,MnC3pTiD,cmC2p  
TjD,GnC3pT2D,KAAK,GmC2pTzD,SAAS,OAAT,CnC3pToE,KAAK,IAAf,C;;QmC6pTrD,OAAO,G;O;KAhBX,  
C;mFAmBA,yB;MnBr2SA,+B;MmBq2SA,sC;QAaoB,Q;QADhB,UnBt2SqC,eAAW,oBmBs2S/B,CnBt2S+B,CAA  
X,C;QmBu2SrB,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,MnB3qTmD,emB2qTnD,GnB3qT8D,KAAK,  
KmB2qT5D,SAAS,OAAT,CnB3qTuE,KAAK,CAAhB,C;;QmB6qTvD,OAAO,G;O;KAhBX,C;mFAmBA,yB;MnB  
x3SA,+B;MmBw3SA,sC;QAaoB,Q;QADhB,UnBz3SqC,eAAW,oBmBy3S/B,CnBz3S+B,CAAX,C;QmB03SrB,2B  
;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,MnB9rTmD,emB8rTnD,GnB9rT8D,KAAK,KmB8rT5D,SAAS,O  
AAT,CnB9rTuE,KAAK,CAAhB,C;;QmBgsTvD,OAAO,G;O;KAhBX,C;mFAmBA,yB;MnB34SA,+B;MmB24SA,  
sC;QAaoB,Q;QADhB,UnB54SqC,eAAW,oBmB44S/B,CnB54S+B,CAAX,C;QmB64SrB,2B;QAAhB,OAAgB,cA  
AhB,C;UAAgB,yB;UACZ,MnBjtTmD,emBjtTnD,GnBjtT8D,KAAK,KmBjtT5D,SAAS,OAAT,CnBjtTuE,KAAK,  
CAAhB,C;;QmBmtTvD,OAAO,G;O;KAhBX,C;mFAmBA,yB;MnB95SA,+B;MmB85SA,sC;QAaoB,Q;QADhB,U  
nB/5SqC,eAAW,oBmB+5S/B,CnB/5S+B,CAAX,C;QmB6gSrB,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ  
,MnBpuTmD,emBouTnD,GnBpuT8D,KAAK,KmBouT5D,SAAS,OAAT,CnBpuTuE,KAAK,CAAhB,C;;QmBsuTv  
D,OAAO,G;O;KAhBX,C;IAmBA,kC;MA2DI,WpBnnTO,MAAO,KoBmnTG,cpBnnTH,EoBikTH,KAkDkB,OpBn  
nTf,C;MoBonTd,WAAW,iBAaA,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WArDqB,GAqD  
P,sBAAK,CAAL,CArDO,EAAnB,KAqDqB,CAAM,CAAN,CArDF,CAqDrB,C;;MArDT,OAuDO,I;K;IApDX,kC;  
MAkEI,WpBtoTO,MAAO,KoBsoTG,cpBtoTH,EoB6kTH,KAyDkB,OpBtoTf,C;MoBuoTd,WAAW,iBAaA,IAAb,  
C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WA5DqB,GA4DP,sBAAK,CAAL,CA5DO,EAAnB,KA4  
DqB,CAAM,CAAN,CA5DF,CA4DrB,C;;MA5DT,OA8DO,I;K;IA3DX,kC;MAyEI,WpBzpTO,MAAO,KoBypTG,c  
pBzpTH,EoBylTH,KAgEkB,OpBzpTf,C;MoB0pTd,WAAW,iBAaA,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAA  
IB,M;QACI,IAAK,WAnEqB,GAmEP,sBAAK,CAAL,CAnEO,EAAnB,KAmEqB,CAAM,CAAN,CAnEF,CAmErB,  
C;;MANET,OAqEO,I;K;IAIEX,kC;MAGFI,WpB5qTO,MAAO,KoB4qTG,cpB5qTH,EoBqmTH,KAuEkB,OpB5qTf  
,C;MoB6qTd,WAAW,iBAaA,IAAb,C;MACX,aAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WA1EqB,GA0EP,s  
BAAK,CAAL,CA1EO,EAAnB,KA0EqB,CAAM,CAAN,CA1EF,CA0ErB,C;;MA1ET,OA4EO,I;K;+EAzEX,yB;M  
AAA,gE;MpB9mTA,iB;MoB8mTA,8C;QAWI,WpBnnTO,MAAO,KoBmnTG,cpBnnTH,EoBmnTS,KAAM,OpBn  
nTf,C;QoBonTd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,sBA  
AK,CAAL,CAAV,EAAmB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAhBX,C;+EAmBA,yB;MAAA  
,gE;MpBjoTA,iB;MoBioTA,8C;QAWI,WpBtoTO,MAAO,KoBsoTG,cpBtoTH,EoBsoTS,KAAM,OpBtoTf,C;QoB  
uoTd,WAAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,sBAAK,CAAL,  
CAAV,EAAmB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAhBX,C;+EAmBA,yB;MAAA,gE;MpBp  
pTA,iB;MoBopTA,8C;QAWI,WpBzpTO,MAAO,KoBypTG,cpBzpTH,EoBypTS,KAAM,OpBzpTf,C;QoB0pTd,W  
AAW,eAAa,IAAb,C;QACX,aAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,sBAAK,CAAL,CAAV,  
EAAmB,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAhBX,C;+EAmBA,yB;MAAA,gE;MpBvqTA,iB;  
MoBuqTA,8C;QAWI,WpB5qTO,MAAO,KoB4qTG,cpB5qTH,EoB4qTS,KAAM,OpB5qTf,C;QoB6qTd,WAAW,e

AAa,IAAb,C;QACX,AAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,sBAAK,CAAL,CAAV,EAAM  
B,MAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAhBX,C;IAmBA,kC;MA8DoB,gB;MAHhB,gBAAGB,c;  
MACHB,WAAW,iBpBhvTJ,MAAO,KoBgvTsB,wBAnDzB,KAmDyB,EAawB,EAaxB,CpBhvTtB,EoBgvTmD,S  
pBhvTnD,CoBgvTH,C;MACX,QAAQ,C;MACQ,OArDL,KAqDK,W;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QA  
CZ,IAAI,KAAK,SAAT,C;UAAoB,K;QACpB,IAAK,WAvDqB,GAuDP,uBAAK,UAAAL,EAakB,BAAL,UAvDO,  
EAuDI,OAvDJ,CAuDrB,C;;MAvDT,OAYDO,I;K;IAtdX,kC;MAuEoB,gB;MAHhB,gBAAGB,c;MACHB,WAAW,i  
BpBrwTJ,MAAO,KoBqwTsB,wBA5DzB,KA4DyB,EAawB,EAaxB,CpBrwTtB,EoBqwTmD,SpBrwTnD,CoBqw  
TH,C;MACX,QAAQ,C;MACQ,OA9DL,KA8DK,W;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,IAAI,KAAK,  
SAAT,C;UAAoB,K;QACpB,IAAK,WAhEqB,GAgEP,uBAAK,UAAAL,EAakB,BAAL,UAhEO,EAgEI,OAhEJ,CA  
gErB,C;;MAhET,OAKEO,I;K;IA/DX,kC;MAGFoB,gB;MAHhB,gBAAGB,c;MACHB,WAAW,iBpB1xTJ,MAAO,K  
oB0xTsB,wBArEzB,KAqEyB,EAawB,EAaxB,CpB1xTtB,EoB0xTmD,SpB1xTnD,CoB0xTH,C;MACX,QAAQ,C  
;MACQ,OAvEL,KAuEK,W;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,IAAI,KAAK,SAAT,C;UAAoB,K;QA  
CpB,IAAK,WazEqB,GAyEP,uBAAK,UAAAL,EAakB,BAAL,UazEO,EAYEI,OAzEJ,CAyErB,C;;MAzET,OA2E  
O,I;K;IAxEX,kC;MAYFoB,gB;MAHhB,gBAAGB,c;MACHB,WAAW,iBpB/yTJ,MAAO,KoB+yTsB,wBA9EzB,K  
A8EyB,EAawB,EAaxB,CpB/yTtB,EoB+yTmD,SpB/yTnD,CoB+yTH,C;MACX,QAAQ,C;MACQ,OAhFL,KAgF  
K,W;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,IAAI,KAAK,SAAT,C;UAAoB,K;QACpB,IAAK,WAlFqB,G  
AkFP,uBAAK,UAAAL,EAakB,BAAL,UAlFO,EAkFI,OAlFJ,CAkFrB,C;;MAIFT,OAoFO,I;K;+EAjFX,yB;MAAA,  
kF;MAAA,gE;MpB1uTA,iB;MoB0uTA,8C;QacoB,UAEY,M;QAL5B,gBAAGB,c;QACHB,WAAW,epBhvTJ,MA  
AO,KoBgvTsB,wBAAN,KAAM,EAawB,EAaxB,CpBhvTtB,EoBgvTmD,SpBhvTnD,CoBgvTH,C;QACX,QAAQ  
,C;QACQ,uB;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,IAAI,KAAK,SAAT,C;YAAoB,K;UACpB,IAAK,W  
AAI,UAAU,uBAAK,UAAAL,EAakB,BAAL,UAAV,EAaqB,OAArB,CAAJ,C;;QAET,OAAO,I;O;KAIBX,C;+EAq  
BA,yB;MAAA,kF;MAAA,gE;MpB/vTA,iB;MoB+vTA,8C;QacoB,UAEY,M;QAL5B,gBAAGB,c;QACHB,WAAW  
,epBrwTJ,MAAO,KoBqwTsB,wBAAN,KAAM,EAawB,EAaxB,CpBrwTtB,EoBqwTmD,SpBrwTnD,CoBqwTH,  
C;QACX,QAAQ,C;QACQ,uB;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,IAAI,KAAK,SAAT,C;YAAoB,K;U  
ACpB,IAAK,WAAI,UAAU,uBAAK,UAAAL,EAakB,BAAL,UAAV,EAaqB,OAArB,CAAJ,C;;QAET,OAAO,I;O;  
KAIBX,C;+EAqBA,yB;MAAA,kF;MAAA,gE;MpBpxTA,iB;MoBoxTA,8C;QacoB,UAEY,M;QAL5B,gBAAGB,c;  
QACHB,WAAW,epB1xTJ,MAAO,KoB0xTsB,wBAAN,KAAM,EAawB,EAaxB,CpB1xTtB,EoB0xTmD,SpB1xT  
nD,CoB0xTH,C;QACX,QAAQ,C;QACQ,uB;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,IAAI,KAAK,SAAT,  
C;YAAoB,K;UACpB,IAAK,WAAI,UAAU,uBAAK,UAAAL,EAakB,BAAL,UAAV,EAaqB,OAArB,CAAJ,C;;QA  
ET,OAAO,I;O;KAIBX,C;8EAqBA,yB;MAAA,kF;MAAA,gE;MpBzyTA,iB;MoByyTA,8C;QacoB,UAEY,M;QAL  
5B,gBAAGB,c;QACHB,WAAW,epB/yTJ,MAAO,KoB+yTsB,wBAAN,KAAM,EAawB,EAaxB,CpB/yTtB,EoB+y  
TmD,SpB/yTnD,CoB+yTH,C;QACX,QAAQ,C;QACQ,uB;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,IAAI,K  
AAK,SAAT,C;YAAoB,K;UACpB,IAAK,WAAI,UAAU,uBAAK,UAAAL,EAakB,BAAL,UAAV,EAaqB,OAArB,  
CAAJ,C;;QAET,OAAO,I;O;KAIBX,C;IAqBA,kC;MA2DI,WpBn3TO,MAAO,KoBm3TG,cpBn3TH,EoBi0TH,KA  
kDkB,KpBn3Tf,C;MoBo3Td,WAAW,iBAaA,IAAb,C;MACX,AAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WAr  
DqB,GAqDP,sBAAK,CAAL,CArDO,EAAnB,KAqDqB,aAAM,CAAN,CArDF,CAqDrB,C;;MArDT,OAuDO,I;K;I  
ApDX,kC;MAkEI,WpBt4TO,MAAO,KoBs4TG,cpBt4TH,EoB60TH,KAyDkB,KpBt4Tf,C;MoBu4Td,WAAW,iBA  
Aa,IAAb,C;MACX,AAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WA5DqB,GA4DP,sBAAK,CAAL,CA5DO,EA  
AnB,KA4DqB,aAAM,CAAN,CA5DF,CA4DrB,C;;MA5DT,OA8DO,I;K;IA3DX,kC;MAYEI,WpBz5TO,MAAO,K  
oBy5TG,cpBz5TH,EoBy1TH,KAgEkB,KpBz5Tf,C;MoB05Td,WAAW,iBAaA,IAAb,C;MACX,AAAU,CAAV,MA  
AkB,IAAIB,M;QACI,IAAK,WAnEqB,GAmEP,sBAAK,CAAL,CAnEO,EAAnB,KAmEqB,aAAM,CAAN,CAnEF,  
CAmErB,C;;MAnET,OAqEO,I;K;IAIEX,kC;MAGFI,WpB56TO,MAAO,KoB46TG,cpB56TH,EoBq2TH,KAuEkB,  
KpB56Tf,C;MoB66Td,WAAW,iBAaA,IAAb,C;MACX,AAAU,CAAV,MAAkB,IAAIB,M;QACI,IAAK,WAlEqB,  
GA0EP,sBAAK,CAAL,CA1EO,EAAnB,KA0EqB,aAAM,CAAN,CA1EF,CA0ErB,C;;MA1ET,OA4EO,I;K;+EAzE  
X,yB;MAAA,gE;MpB92TA,iB;MoB82TA,8C;QAWI,WpBn3TO,MAAO,KoBm3TG,cpBn3TH,EoBm3TS,KAAM,  
KpBn3Tf,C;QoBo3Td,WAAW,eAAa,IAAb,C;QACX,AAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAA  
U,sBAAK,CAAL,CAAV,EAAMB,kBAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAhBX,C;+EAmbA,yB;  
MAAA,gE;MpBj4TA,iB;MoBi4TA,8C;QAWI,WpBt4TO,MAAO,KoBs4TG,cpBt4TH,EoBs4TS,KAAM,KpBt4Tf,

C;QoBu4Td,WAAW,eAAa,IAAb,C;QACX,AAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,sBAAK,CAAL,CAAV,EAAMb,kBAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAhBX,C;+EAmBA,yB;MAAA,gE;MpBp5TA,iB;MoBo5TA,8C;QAWI,WpBz5TO,MAAO,KoBy5TG,cpBz5TH,EoBy5TS,KAAM,KpBz5Tf,C;QoB05Td,WAAW,eAAa,IAAb,C;QACX,AAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,sBAAK,CAAL,CAAV,EAAMb,kBAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAhBX,C;+EAmBA,yB;MAAA,gE;MpBv6TA,iB;MoBu6TA,8C;QAWI,WpB56TO,MAAO,KoB46TG,cpB56TH,EoB46TS,KAAM,KpB56Tf,C;QoB66Td,WAAW,eAAa,IAAb,C;QACX,AAAU,CAAV,MAAkB,IAAIB,M;UACI,IAAK,WAAI,UAAU,sBAAK,CAAL,CAAV,EAAMb,kBAAM,CAAN,CAAnB,CAAJ,C;;QAET,OAAO,I;O;KAhBX,C;IAmBA,2B;MAQoB,Q;MADhB,UAAgB,W;MACHb,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,MnCjnUiD,SmCinUjD,GnCjnU2D,KAAK,GmCinUzD,OnCjnUoE,KAAx,IAAf,C;;MmCmnUrD,OAAO,G;K;IAGX,2B;MAQoB,Q;MADhB,UAAiB,2B;MACjB,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,MnB5nUmD,UmB4nUnD,GnB5nU8D,KAAK,KmB4nU5D,OnB5nUuE,KAAx,CAAhB,C;;MmB8nUvD,OAAO,G;K;IAGX,2B;MAQoB,Q;MADhB,UAAgB,W;MACHb,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,MnC7oUiD,SmC6oUjD,GnC7oU2D,KAAK,GAAW,CD2O5C,SoCk6TxB,OpCl6TkC,KAAL,GAAiB,GAAtB,CC3O4C,MAAX,IAAf,C;;MmC+oUrD,OAAO,G;K;IAGX,2B;MAQoB,Q;MADhB,UAAgB,W;MACHb,wBAAgB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QACI,MnC3pUiD,SmC2pUjD,GnC3pU2D,KAAK,GAAW,CC4O5C,SkC+6TxB,OIC/6TkC,KAAL,GAAiB,KAAtB,CD5O4C,MAAX,IAAf,C;;MmC6pUrD,OAAO,G;K;+EAGX,yB;MAAA,0C;MnC2TA,6B;MmCw2TA,4B;QAOI,OnCr2TmC,cmCq2TpB,IAAR,iBAAQ,CnCr2ToB,C;O;KmC81TvC,C;+EAUA,yB;MAAA,0C;MnBn2TA,+B;MmBm2TA,4B;QAOI,OnBh2TsC,emBg2TvB,IAAR,iBAAQ,CnBh2TuB,C;O;KmBy1T1C,C;+EAUA,yB;MAAA,sC;MnC53TA,6B;MmC43TA,iBAOiB,yB;QpCz9Tb,6B;eoCy9Ta,c;UAAE,OpCh9ToB,coCg9TpB,EpCh9T8B,KAAL,GAAiB,GAAtB,C;S;OoCg9TtB,C;MAPjB,4B;QA7iBoB,Q;QADhB,UnCp0SmC,cmCo0SnB,CnCp0SmB,C;QmCq0SnB,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,MnCxoTiD,cmCwoTjD,GnCxoT2D,KAAK,GAAW,CD2O5C,coC65Sf,OpC75SyB,KAAL,GAAiB,GAAtB,CC3O4C,MAAX,IAAf,C;;QmC2rUrD,OAJjBO,G;O;KA0iBX,C;+EAUA,yB;MAAA,sC;MnCt4TA,6B;MmCs4TA,iBAOiB,yB;QlCl+Tb,6B;ekCk+Ta,c;UAAE,OICz9ToB,ckCy9TpB,EICz9T8B,KAAAL,GAAiB,KAAtB,C;S;OkCy9TtB,C;MAPjB,4B;QApiBoB,Q;QADhB,UnCv1SmC,cmCu1SnB,CnCv1SmB,C;QmCw1SnB,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UACZ,MnC3pTiD,cmC2pTjD,GnC3pT2D,KAAK,GAAW,CC4O5C,ckC+6Sf,OIC/6SyB,KAAL,GAAiB,KAAtB,CD5O4C,MAAX,IAAf,C;;QmCqsUrD,OAXiBO,G;O;KaiiBX,C;IC3vUA,mC;MAQoB,UACL,M;MAHX,aAAa,gBAAW,cAAX,C;MACb,YAAY,C;MACI,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,oBAAO,cAAP,EAAO,sBAAP,WAAkB,OAAIB,C;;MACJ,OAAO,M;K;IAGX,kC;MAQoB,UACL,M;MAHX,aAAa,eAAU,cAAV,C;MACb,YAAY,C;MACI,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,oBAAO,cAAP,EAAO,sBAAP,WAAkB,OAAIB,C;;MACJ,OAAO,M;K;IAGX,mC;MAQoB,UACL,M;MAHX,aAAa,iBAAAY,cAAZ,C;MACb,YAAY,C;MACI,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,oBAAO,cAAP,EAAO,sBAAP,WAAkB,OAAIB,C;;MACJ,OAAO,M;K;IAGX,2B;MAQoB,Q;MADhB,UAAgB,W;MACA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,MpCAiD,SoCAjD,GpCA2D,KAAK,GoCAzD,OpCAoE,KAAx,IAAf,C;;MocErD,OAAO,G;K;IAGX,2B;MAQoB,Q;MADhB,UAAiB,2B;MACD,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,MpBXmD,UoBwND,GpBX8D,KAAK,KoBW5D,OpBXuE,KAAx,CAAhB,C;;MoBavD,OAAO,G;K;IAGX,2B;MAQoB,Q;MADhB,UAAgB,W;MACA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,MpC5BiD,SoC4BjD,GpC5B2D,KAAK,GAAW,CD2O5C,SqC/MxB,OrC+MkC,KAAL,GAAiB,GAAtB,CC3O4C,MAAX,IAAf,C;;MoC8BrD,OAAO,G;K;IAGX,2B;MAQoB,Q;MADhB,UAAgB,W;MACA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,MpC1CiD,SoC0CjD,GpC1C2D,KAAK,GAAW,CC4O5C,SmClMxB,OnCkMkC,KAAL,GAAiB,KAAtB,CD5O4C,MAAX,IAAf,C;;MoC4CrD,OAAO,G;K;IC3GX,wB;MAMI,OrCuCkE,YqCvCvD,CrCuCwE,KAAjB,EqCvClD,CrCuC+E,KAA7B,CqCvCvD,KAAJ,GAAY,CAAZ,GAAmB,C;K;IAG9B,wB;MAMI,OrBsCmE,aqBtCxD,CrBsC0E,KAAIB,EqBtCnD,CrBsCiF,KAA9B,CqBtCxD,KAAJ,GAAY,CAAZ,GAAmB,C;K;IAG9B,wB;MAMI,OtCkGf,0BsCLrE,CtCgP2B,KAAL,GAAiB,GA3O8B,EsCLhE,CtCgPsB,KAAL,GAAiB,GA3O8B,CsCLrE,KAAJ,GAAY,CAAZ,GAAmB,C;K;IAG9B,wB;MAMI,OpClIf,0BoCjE,CpCwO2B,KAAL,GAAiB,KApO+B,EoCjJE,CpCwOsB,KAAL,GAAiB,KApO+B,CoCjTE,KAAJ,GAAY,CAAZ,GAAmB,C;K;mFAG9B,yB;MAAA,8C;MAAA,0



A6B,CAAC,IAAD,IAA1D,C;K;IAG3B,iC;MAMI,oBAAoB,kBAAO,CAA3B,EAA8B,IAA9B,C;MACA,OAAO,w  
CAAiB,yBAAgB,eAAhB,EAAuB,cAAvB,EAAiC,SAAK,KAAL,cAAy,CAAhB,GAAMb,IAAnB,GAA8B,IAAD,  
aAA1D,C;K;IAG5B,iC;MAQI,IvC/OgF,0BuC+O5E,EvCjKc,KAAL,GAAiB,GA3O8B,EuC+OtE,6BAAM,UvCJsB  
,KAAL,GAAiB,GA3O8B,CuC+O5E,KAAJ,C;QAA2B,OAAO,iCAAU,M;MACHc,WvC6BuB,SuC7B5B,SvC6Bs  
C,KAAL,GAAiB,GAAtB,C;MuC7BV,YAAK,W;MAA9B,OtCjD6D,oBAhJP,SAAU,CD8N7B,SuC7BV,EvC6BoB  
,KAAL,GAAiB,GAAtB,CC9N6B,MAAK,GDAK,KCAO,KAAZ,IAAf,CAGJO,C;K;IsCoDjE,iC;MAQI,ItC3OkE,Y  
sC2O9D,EtC3O+E,KAAjB,EsC2OxD,4BAAK,UtC3OgF,KAA7B,CsC2O9D,KAAJ,C;QAA0B,OAAO,iCAAU,M;  
MAC3C,OtC7D6D,csC6DtD,StC7DsD,EAhJP,SsC6MtC,EtC7MgD,KAAK,GAAY,CsC6M5D,WtC7M4D,MAAZ,  
IAAf,CAGJO,C;K;IsCgEjE,iC;MAQI,ItB/OmE,asB+O/D,EtB/OiF,KAAiB,EsB+OzD,6BAAM,UtB/OiF,KAA9B,Cs  
B+O/D,KAAJ,C;QAA2B,OAAO,kCAAW,M;MAC7C,OtBzE+D,iBsByExD,StBzEwD,EA7IP,UsBsNxC,EtBtNmD  
,KAAK,UAAy,ChByP/C,UAAW,oBAAL,CsCnCb,WtCmCsB,MAAK,CAAL,iBAAN,CgBzP+C,MAAZ,CAAhB  
,CA6IO,C;K;IsB4EnE,iC;MAQI,IrC3QiF,0BqC2Q7E,ErCvCkC,KAAL,GAAiB,KApO+B,EqC2QvE,8BAAO,UrC  
vCqB,KAAL,GAAiB,KApO+B,CqC2Q7E,KAAJ,C;QAA4B,OAAO,iCAAU,M;MACjC,WrcNuB,SqCM5B,SrCnS  
C,KAAL,GAAiB,KAAtB,C;MqCMV,YAAK,W;MAA9B,OtCrF6D,oBAhJP,SAAU,CC+N7B,SqCMV,ErCnOb,K  
AAL,GAAiB,KAAtB,CD/N6B,MAAK,GCAK,KDAO,KAAZ,IAAf,CAGJO,C;K;IsCwFjE,kD;MAUI,OtCjRkE,Ys  
CiRvD,StCjRwE,KAAjB,EsCiRhD,YtCjR6E,KAA7B,CsCiRvD,IAAJ,GAAY,YAAzB,GAA2C,S;K;IAGtD,kD;M  
AUI,OtBtRmE,asBsRxD,StBtR0E,KAAiB,EsBsRjD,YtBtR+E,KAA9B,CsBsRxD,IAAJ,GAAY,YAAzB,GAA2C,  
S;K;IAGtD,kD;MAUI,OvC3TgF,0BuC2TrE,SvChF2B,KAAL,GAAiB,GA3O8B,EuC2T9D,YvChFoB,KAAL,GA  
AiB,GA3O8B,CuC2TrE,IAAJ,GAAY,YAAzB,GAA2C,S;K;IAGtD,kD;MAUI,OrChUiF,0BqCgUtE,SrC5F2B,KA  
AL,GAAiB,KApO+B,EqCgU/D,YrC5FoB,KAAL,GAAiB,KApO+B,CqCgUtE,IAAJ,GAAY,YAAzB,GAA2C,S;  
K;IAGtD,iD;MAUI,OtCrUkE,YsCqUvD,StCrUwE,KAAjB,EsCqUhD,YtCrU6E,KAA7B,CsCqUvD,IAAJ,GAAY,  
YAAzB,GAA2C,S;K;IAGtD,iD;MAUI,OtB1UmE,asB0UxD,StB1U0E,KAAiB,EsB0UjD,YtB1U+E,KAA9B,CsB0  
UxD,IAAJ,GAAY,YAAzB,GAA2C,S;K;IAGtD,iD;MAUI,OvC/WgF,0BuC+WrE,SvCpI2B,KAAL,GAAiB,GA3  
O8B,EuC+W9D,YvCpIoB,KAAL,GAAiB,GA3O8B,CuC+WrE,IAAJ,GAAY,YAAzB,GAA2C,S;K;IAGtD,iD;M  
AUI,OrCpXiF,0BqCoXtE,SrChJ2B,KAAL,GAAiB,KApO+B,EqCoX/D,YrChJoB,KAAL,GAAiB,KApO+B,CqCo  
XtE,IAAJ,GAAY,YAAzB,GAA2C,S;K;IAGtD,4D;MAUI,ItCzXkE,YsCyX9D,YtCzX+E,KAAjB,EsCyX/C,YtCz  
X4E,KAA7B,CsCyX9D,IAAJ,C;QAAiC,MAAM,gCAAYB,oDAAiD,YAAjD,8BAAoF,YAApF,MAAZB,C;MACv  
C,ItC1XkE,YsC0X9D,StC1X+E,KAAjB,EsC0XvD,YtC1XoF,KAA7B,CsC0X9D,IAAJ,C;QAAyB,OAAO,Y;MAC  
hC,ItC3XkE,YsC2X9D,StC3X+E,KAAjB,EsC2XvD,YtC3XoF,KAA7B,CsC2X9D,IAAJ,C;QAAyB,OAAO,Y;MA  
ChC,OAAO,S;K;IAGX,4D;MAUI,ItBjYmE,asBiY/D,YtBjYiF,KAAiB,EsBiYhD,YtBjY8E,KAA9B,CsBiY/D,IAA  
J,C;QAAiC,MAAM,gCAAYB,oDAAiD,YAAjD,8BAAoF,YAApF,MAAZB,C;MACvC,ItBiYmE,asBkY/D,StBiYiF  
,KAAiB,EsBkYxD,YtBiYsF,KAA9B,CsBkY/D,IAAJ,C;QAAyB,OAAO,Y;MACHc,ItBnYmE,asBmY/D,StBnYiF,  
KAAiB,EsBmYxD,YtBnYsF,KAA9B,CsBmY/D,IAAJ,C;QAAyB,OAAO,Y;MACHc,OAAO,S;K;IAGX,4D;MAU  
I,IvCzAgF,0BuCya5E,YvC9LkC,KAAL,GAAiB,GA3O8B,EuCya7D,YvC9LmB,KAAL,GAAiB,GA3O8B,CuCya5  
E,IAAJ,C;QAAiC,MAAM,gCAAYB,oDAAiD,YAAjD,8BAAoF,YAApF,MAAZB,C;MACvC,IvC1agF,0BuC0a5E,  
SvC/LkC,KAAL,GAAiB,GA3O8B,EuC0arE,YvC/L2B,KAAL,GAAiB,GA3O8B,CuC0a5E,IAAJ,C;QAAyB,OAA  
O,Y;MACHc,IvC3agF,0BuC2a5E,SvChMkC,KAAL,GAAiB,GA3O8B,EuC2arE,YvChM2B,KAAL,GAAiB,GA3O  
8B,CuC2a5E,IAAJ,C;QAAyB,OAAO,Y;MACHc,OAAO,S;K;IAGX,4D;MAUI,IrCjbiF,0BqCib7E,YrC7MkC,KA  
AL,GAAiB,KApO+B,EqCib9D,YrC7MmB,KAAL,GAAiB,KApO+B,CqCib7E,IAAJ,C;QAAiC,MAAM,gCAAYB,  
oDAAiD,YAAjD,8BAAoF,YAApF,MAAZB,C;MACvC,IrClbiF,0BqCkb7E,SrC9MkC,KAAL,GAAiB,KApO+B,E  
qCkbtE,YrC9M2B,KAAL,GAAiB,KApO+B,CqCkb7E,IAAJ,C;QAAyB,OAAO,Y;MACHc,IrCnbiF,0BqCmb7E,Sr  
C/MkC,KAAL,GAAiB,KApO+B,EqCmbtE,YrC/M2B,KAAL,GAAiB,KApO+B,CqCmb7E,IAAJ,C;QAAyB,OAA  
O,Y;MACHc,OAAO,S;K;IAGX,uC;MAcW,Q;MAJP,IAAI,8CAAJ,C;QACI,OAAy,WAAL,SAAK,EAAe,KAAf,C  
;OAEhB,IAAI,KAAM,UAAV,C;QAAqB,MAAM,gCAAYB,4CAAyC,KAAzC,MAAZB,C;MAEvB,ItC9b8D,YsC8  
b9D,StC9b+E,KAAjB,EsC8bvD,KAAM,MtC9b8E,KAA7B,CsC8b9D,K;QAA4B,OAAN,KAAM,M;;QAC5B,ItC/b  
8D,YsC+b9D,StC/b+E,KAAjB,EsC+bvD,KAAM,atC/b8E,KAA7B,CsC+b9D,K;UAAmC,OAAN,KAAM,a;;UAC3  
B,gB;;MAHZ,W;K;IAOJ,uC;MAcW,Q;MAJP,IAAI,8CAAJ,C;QACI,OAAy,WAAL,SAAK,EAAgB,KAAhB,C;O  
AEhB,IAAI,KAAM,UAAV,C;QAAqB,MAAM,gCAAYB,4CAAyC,KAAzC,MAAZB,C;MAEvB,ItB3c+D,asB2c/D

,StB3ciF,KAAIB,EsB2cxD,KAAM,MtB3cgF,KAA9B,CsB2c/D,K;QAA4B,OAAN,KAAM,M;;QAC5B,ItB5c+D,as  
B4c/D,StB5ciF,KAAIB,EsB4cxD,KAAM,atB5cgF,KAA9B,CsB4c/D,K;UAAmC,OAAN,KAAM,a;;UAC3B,gB;;M  
AHZ,W;K;IC/fJ,2B;MAUoB,Q;MADhB,UAAgB,W;MACA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,M  
vCoDiD,SuCPdJd,GvCoD2D,KAAK,GuCpDzD,OvCoDoE,KAAX,IAAf,C;;MuClDrD,OAAO,G;K;IAGX,2B;MA  
UoB,Q;MADhB,UAAiB,2B;MACD,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,MvBuCmD,UuBvCnD,Gv  
BuC8D,KAAK,KuBvC5D,OvBuCuE,KAAX,CAAhB,C;;MuBrCvD,OAAO,G;K;IAGX,2B;MAUoB,Q;MADhB,U  
AAgB,W;MACA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,MvCoBiD,SuCPbJd,GvCoB2D,KAAK,GA  
AW,CD2O5C,SwC/PxB,OxC+PkC,KAAL,GAAiB,GAAtB,CC3O4C,MAAX,IAAf,C;;MuClBrD,OAAO,G;K;IAG  
X,2B;MAUoB,Q;MADhB,UAAgB,W;MACA,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACZ,MvClid,SuClJd  
,GvCl2D,KAAK,GAAW,CC4O5C,SsChPxB,OtCgPkC,KAAL,GAAiB,KAAtB,CD5O4C,MAAX,IAAf,C;;MuCFr  
D,OAAO,G;K;,,,,;ICuCP,iD;MAAA,qE;MAAgB,4B;MANpB,uC;MAMI,Y;K;IACA,4D;MAAA,qE;MAAgC,wBA  
AM,OAAN,Q;MAPpC,uC;MAOI,Y;K;IACA,mE;MAAA,qE;MAAmD,6BAAM,OAAN,EAAe,KAAf,C;MARvD,u  
C;MAQI,Y;K;IACA,0D;MAAA,qE;MAAiC,wBAAM,KAAN,Q;MATrC,uC;MASI,Y;K;ICxGJ,gC;K;,,,,;ICuBoC,w  
C;8BAAsC,O;K;,,,,,;yCC0rE,6B;MASI,MAAM,yB;K;,,,,;0CAyDV,sB;MASI,OAAO,I;K;,,,,;  
,,,,,;ICnYf,wB;K;kCAEI,Y;MAA4B,sB;K;;IAMhC,wB;K;kCAEI,Y;MAA4B,mC;K;;IAMhC,yB;K;mCAEI,Y  
;MAA4B,uB;K;;IAMhC,uB;K;iCAEI,Y;MAA4B,qB;K;;IAMhC,wB;K;kCAEI,Y;MAA4B,sB;K;;IAMhC,yB;K;mC  
AEI,Y;MAA4B,uB;K;;IAMhC,0B;K;oCAEI,Y;MAA4B,wB;K;;IAMhC,2B;K;qCAEI,Y;MAA4B,yB;K;;ICtDM,oD  
;MAA2C,uB;MAAjB,gB;MAC5D,sBAAGC,InBkCU,I;MmBjC1C,iBAAmC,YAAO,CAAX,GAAC,SAAS,IAAvB,  
GAAiC,SAAS,I;MACzE,cAA4B,cAA5B,GAAqC,KnBgCK,ImBhC1C,GAAqD,mB;K;gDAErD,Y;MAAkC,qB;K;i  
DAEIC,Y;MACI,YAAAY,W;MACZ,IAAI,UAAS,mBAAb,C;QACI,IAAI,CAAC,cAAL,C;UAAc,MAAa,6B;QAC3  
B,iBAAU,K;;QAGV,4BAAQ,SAAR,I;;MAEJ,OAAa,OAAN,KAAM,C;K;;IAQgB,mD;MAAyC,sB;MAAjB,gB;M  
ACzD,sBAAGC,I;MACHC,iBAAmC,YAAO,CAAX,GAAC,SAAS,IAAvB,GAAiC,SAAS,I;MACzE,cAA4B,cAAJ,  
GAAa,KAAb,GAAwB,mB;K;+CAEhD,Y;MAAkC,qB;K;+CAEIC,Y;MACI,YAAAY,W;MACZ,IAAI,UAAS,mBA  
Ab,C;QACI,IAAI,CAAC,cAAL,C;UAAc,MAAa,6B;QAC3B,iBAAU,K;;QAGV,4BAAQ,SAAR,I;;MAEJ,OAAO,  
K;K;;IAQuB,oD;MAA4C,uB;MAAIB,gB;MAC5D,sBAAiC,I;MACjC,iBAAmC,uBAAO,CAAX,GAAC,sBAAS,IA  
AT,MAAd,GAAiC,sBAAS,IAAT,M;MACHC,cAA6B,cAAJ,GAAa,KAAb,GAAwB,mB;K;gDAEjD,Y;MAAkC,qB  
;K;iDAEIC,Y;MACI,YAAAY,W;MACZ,IAAI,cAAS,mBAAT,CAAJ,C;QACI,IAAI,CAAC,cAAL,C;UAAc,MAAa,  
6B;QAC3B,iBAAU,K;;QAGV,8BAAQ,SAAR,C;;MAEJ,OAAO,K;K;;IC9DX,oD;MA6CA,uC;MATCI,IAAI,SAA  
Q,CAAZ,C;QAAe,MAAa,gCAAYB,wBAAzB,C;MAC5B,IAAI,SAAQ,WAAZ,C;QAA2B,MAAa,gCAAYB,wEAA  
zB,C;MAG5C,aAGyB,K;MAEZB,YAGuF,OAA/D,0BAA0B,KpBcR,IoBdlB,EAAc,YpBcpB,IoBdlB,EAAyD,IA  
AzD,CAA+D,C;MAEvF,YAGuB,I;K;yCAEvB,Y;MAAwC,mCAAwB,UAAxB,EAA+B,SAA/B,EAAqC,SAAR,C,C  
;K;wCAExC,Y;MAMqC,OAAI,YAAO,CAAX,GAAC,aAAQ,SAAtB,GAAgC,aAAQ,S;K;uCAE7E,iB;MACI,iDA  
A6B,kBAAa,KAAM,UAAAnB,KAC7B,eAAS,KAAM,MAAf,IAAwB,cAAQ,KAAM,KAAtC,IAA8C,cAAQ,KAA  
M,KAD/B,CAA7B,C;K;yCAGJ,Y;MACI,OAAI,cAAJ,GAAe,EAAf,GAAwB,OAAM,OAAC,UpBRG,IoBQR,UA  
AkB,SpBRV,IoBQR,KAAN,SAAqC,SAAR,C,I;K;yCAE5B,Y;MAAkC,OAAI,YAAO,CAAX,GAAC,oBAAE,UAAF  
,+BAAU,SAAV,eAAqB,SAAnc,GAA8C,oBAAE,UAAF,qCAAgB,SAAhB,gBAA4B,CAAC,SAAD,IAA5B,C;K;I  
AEhF,qC;MAAA,yC;K;kEACI,sC;MAQ2F,2BAAgB,UAAhB,EAA4B,QAA5B,EAAc,IAAtC,C;K;;IAT/F,iD;M  
AAA,gD;QAAA,+B;OAAA,yC;K;;IAiBA,mD;MA6CA,sC;MATCI,IAAI,SAAQ,CAAZ,C;QAAe,MAAa,gCAAYB,  
wBAAzB,C;MAC5B,IAAI,SAAQ,WAAZ,C;QAA2B,MAAa,gCAAYB,wEAAzB,C;MAG5C,aAGwB,K;MAExB,Y  
AGuB,0BAA0B,KAA1B,EAAiC,YAAjC,EAA+C,IAA/C,C;MAEvB,YAGuB,I;K;wCAEvB,Y;MAAuC,kCAAuB,  
UAAvB,EAA8B,SAA9B,EAAoC,SAAP,C,C;K;uCAEvC,Y;MAMqC,OAAI,YAAO,CAAX,GAAC,aAAQ,SAAtB,G  
AAgC,aAAQ,S;K;sCAE7E,iB;MACI,gDAA4B,kBAAa,KAAM,UAAAnB,KAC5B,eAAS,KAAM,MAAf,IAAwB,cA  
AQ,KAAM,KAAtC,IAA8C,cAAQ,KAAM,KADhC,CAA5B,C;K;wCAGJ,Y;MACI,OAAI,cAAJ,GAAe,EAAf,GA  
AwB,OAAM,MAAK,UAAAL,QAAa,SAAb,IAAN,SAA2B,SAA3B,I;K;wCAE5B,Y;MAAkC,OAAI,YAAO,CAAX,  
GAAgB,UAAF,qBAAU,SAAV,cAAqB,SAAnc,GAAgD,UAAF,2BAAgB,SAAhB,eAA4B,CAAC,SAAD,IAA5B,  
C;K;IAEhF,oC;MAAA,wC;K;iEACI,sC;MAQwF,0BAAe,UAAf,EAA2B,QAA3B,EAAqC,IAArC,C;K;;IAT5F,gD  
;MAAA,+C;QAAA,8B;OAAA,wC;K;;IAiBA,oD;MA6CA,uC;MATCI,IAAI,gBAAJ,C;QAAgB,MAAa,gCAAYB,w  
BAAzB,C;MAC7B,IAAI,sCAAJ,C;QAA4B,MAAa,gCAAYB,yEAAzB,C;MAG7C,aAGyB,K;MAEZB,YAGwB,4B

AA0B,KAA1B,EAAiC,YAAjC,EAA+C,IAA/C,C;MAExB,YAGwB,I;K;yCAExB,Y;MAAwC,mCAAwB,UAAxB,  
EAA+B,SAA/B,EAAqC,SAArC,C;K;wCAExC,Y;MAMqC,OAAI,uBAAO,CAAX,GAAC,2BAAQ,SAAR,KAAf,  
GAAgC,2BAAQ,SAAR,K;K;uCAErE,iB;MACI,iDAA6B,kBAAa,KAAM,UAAAnB,KAC7B,mBAAS,KAAM,MAA  
f,KAAwB,kBAAQ,KAAM,KAAf,CAAxB,IAA8C,kBAAQ,KAAM,KAAf,CADjB,CAA7B,C;K;yCAGJ,Y;MACI,  
OAAI,cAAJ,GAAe,EAAf,GAAwB,iCAAM,iCAAM,eAAW,8BAAW,EAAX,CAAX,CAAN,MAAoC,cAAU,6BA  
AU,EAAV,CAAV,CAAPC,CAAN,MAAuE,cAAU,6BAAU,EAAV,CAAV,CAAvE,CAAIg,Q;K;yCAE7H,Y;MA  
AkC,OAAI,uBAAO,CAAX,GAAgB,UAAf,qBAAU,SAAV,yBAAqB,SAArB,WAAf,GAAgD,UAAf,2BAAgB,S  
AAhB,yBAA6B,SAAD,aAA5B,W;K;IAEHf,qC;MAAA,yC;K;kEACI,sC;MAQ4F,2BAAgB,UAAhB,EAA4B,QA  
A5B,EAAcC,IAAtC,C;K;;;IAThG,iD;MAAA,gD;QAAA,+B;OAAA,yC;K;;;6CCIKa,iB;MAGkD,+BAAS,UAAT,  
UAAkB,wBAAS,iBAAT,M;K;oCAEpE,Y;MAKgC,oCAAQ,iBAAR,K;K;;;I7CpBd,wC;MAkBIB,iC;MATBsD,2BA  
AgB,KAAhB,EAAuB,YAAvB,EAAqC,CAArC,C;K;kFAC7B,Y;MAAQ,8B;K;yFACD,Y;MAAQ,6B;K;2CAExC,i  
B;MAA8C,qBAAS,KAAT,IAAkB,SAAS,S;K;kCAEzE,Y;MAKkC,oBAAQ,S;K;iCAE1C,iB;MACI,2CAAuB,kBA  
Aa,KAAM,UAAAnB,KACvB,eAAS,KAAM,MAAf,IAAwB,cAAQ,KAAM,KADf,CAAvB,C;K;mCAGJ,Y;MACI,O  
AAI,cAAJ,GAAe,EAAf,GAAwB,OAAK,UwBkBS,IxBIBd,UAAkB,SwBkBJ,IxBIBd,K;K;mCAE5B,Y;MAAkC,2  
BAAE,UAAf,+BAAU,SAAV,C;K;IAEIC,+B;MAAA,mC;MACI,aAC8B,cAAY,OAAf,CAAE,CAAZ,EAAwB,O  
AAF,CAAE,CAAxB,C;K;;;IAFIC,2C;MAAA,0C;QAAA,yB;OAAA,mC;K;;;IASiB,uC;MAkBjB,gC;MATBmD,0BA  
Ae,KAAf,EAAcB,YAAtB,EAAoC,CAAPC,C;K;iFAC3B,Y;MAAQ,iB;K;wFACD,Y;MAAQ,gB;K;0CAEvC,iB;M  
AA6C,qBAAS,KAAT,IAAkB,SAAS,S;K;iCAExE,Y;MAKkC,oBAAQ,S;K;gCAE1C,iB;MACI,0CAAsB,kBAaA,  
KAAM,UAAAnB,KACtB,eAAS,KAAM,MAAf,IAAwB,cAAQ,KAAM,KADhB,CAAtB,C;K;kCAGJ,Y;MACI,OA  
AI,cAAJ,GAAe,EAAf,GAAwB,MAAK,UAAAL,QAAa,SAAb,I;K;kCAE5B,Y;MAAkC,OAAE,UAAf,qBAAU,S;K;  
IAE5C,8B;MAAA,kC;MACI,aAC6B,aAAS,CAAT,EAAY,CAAZ,C;K;;;IAFjC,0C;MAAA,yC;QAAA,wB;OAAA,  
kC;K;;;IASkB,wC;MAkBIB,iC;MATBsD,2BAAgB,KAAhB,EAAuB,YAAvB,K;K;kFAC7B,Y;MAAQ,iB;K;yFACD  
,Y;MAAQ,gB;K;2CAExC,iB;MAA8C,kCAAS,KAAT,UAAkB,sBAAS,SAAT,M;K;kCAEHf,Y;MAKkC,kCAAQ,  
SAAR,K;K;iCAEIC,iB;MACI,2CAAuB,kBAAa,KAAM,UAAAnB,KACvB,mBAAS,KAAM,MAAf,KAAwB,kBAA  
Q,KAAM,KAAf,CADD,CAAvB,C;K;mCAGJ,Y;MACI,OAAI,cAAJ,GAAe,EAAf,GAAwB,iCAAM,eAAW,8BA  
AW,EAAX,CAAX,CAAN,MAAoC,cAAU,6BAAU,EAAV,CAAV,CAAPC,CAA8D,Q;K;mCAE1F,Y;MAAkC,O  
AAE,UAAf,qBAAU,SAAV,W;K;IAEIC,+B;MAAA,mC;MACI,aAC8B,qB;K;;;IAFIC,2C;MAAA,0C;QAAA,yB;  
OAAA,mC;K;;;I8C9EJ,gB;MAAA,oB;K;8BAII,Y;MAA0B,oB;K;;;IAJ9B,4B;MAAA,2B;QAAA,U;OAAA,oB;K;I  
CEA,yC;MAAA,e;MAAA,iB;MAAA,uB;K;IAAA,uC;MAAA,0C;O;MAII,kE;MAEA,wF;MAEA,oF;MAEA,wE;  
MAEA,kE;MAEA,oF;MAEA,sF;MAEA,8E;MAEA,wE;MAEA,sF;MAEA,uF;MAEA,iE;MAEA,6E;MAEA,iE;MA  
EA,2E;K;;;IA5BA,8C;MAAA,6B;MAAA,sC;K;;;IAEA,yD;MAAA,6B;MAAA,iD;K;;;IAEA,uD;MAAA,6B;MAAA,  
+C;K;;;IAEA,iD;MAAA,6B;MAAA,yC;K;;;IAEA,8C;MAAA,6B;MAAA,sC;K;;;IAEA,uD;MAAA,6B;MAAA,+C;K  
;;;IAEA,wD;MAAA,6B;MAAA,gD;K;;;IAEA,oD;MAAA,6B;MAAA,4C;K;;;IAEA,iD;MAAA,6B;MAAA,yC;K;;;IA  
EA,wD;MAAA,6B;MAAA,gD;K;;;IAEA,wD;MAAA,6B;MAAA,gD;K;;;IAEA,6C;MAAA,6B;MAAA,qC;K;;;IAEA  
,mD;MAAA,6B;MAAA,2C;K;;;IAEA,6C;MAAA,6B;MAAA,qC;K;;;IAEA,kD;MAAA,6B;MAAA,0C;K;;;IAhCJ,m  
C;MAAA,+oB;K;;;IAAA,wC;MAAA,a;aAAA,O;UAAA,2C;aAAA,kB;UAAA,sD;aAAA,gB;UAAA,oD;aAAA,U;U  
AAA,8C;aAAA,O;UAAA,2C;aAAA,gB;UAAA,oD;aAAA,iB;UAAA,qD;aAAA,a;UAAA,iD;aAAA,U;UAAA,8C;  
aAAA,iB;UAAA,qD;aAAA,iB;UAAA,qD;aAAA,M;UAAA,0C;aAAA,Y;UAAA,gD;aAAA,M;UAAA,0C;aAAA,  
W;UAAA,+C;gBAAA,uE;;K;;IAqCA,4C;MAAA,e;MAAA,iB;MAAA,uB;K;IAAA,0C;MAAA,6C;O;MAMI,0E;M  
AEA,0E;MAEA,4E;K;;;IAJA,kD;MAAA,gC;MAAA,0C;K;;;IAEA,kD;MAAA,gC;MAAA,0C;K;;;IAEA,mD;MAAA  
,gC;MAAA,2C;K;;;IAVJ,sC;MAAA,sI;K;;;IAAA,2C;MAAA,a;aAAA,Q;UAAA,+C;aAAA,Q;UAAA,+C;aAAA,S;U  
AAA,gD;gBAAA,0E;;K;;IAwB8B,gC;MAAC,oC;K;;IAQE,0B;MAAC,qB;QAAA,iD;MAAA,kB;K;;IAEIC,sB;K;;I  
AMA,4B;K;;IC/EA,yB;K;;IAQA,6B;K;;ICnBA,mB;MAEI,UAAU,IAAI,C;MACd,OAAW,OAAO,CAAX,GAAC,G  
AAf,GAAuB,MAAM,CAAN,I;K;IAGIC,qB;MACI,UAAU,SAAI,CAAJ,C;MACV,OAAW,kBAAO,CAAX,GAAC,  
GAAf,GAAuB,QAAM,CAAN,C;K;IAGIC,mC;MAEI,OAAO,IAAI,IAAI,CAAJ,EAAO,CAAP,IAAY,IAAI,CAAJ,  
EAAO,CAAP,CAAZ,IAAJ,EAA2B,CAA3B,C;K;IAGX,qC;MACI,OAAO,MAAI,MAAI,CAAJ,EAAO,CAAP,WA  
AY,MAAI,CAAJ,EAAO,CAAP,CAAZ,CAAJ,EAA2B,CAA3B,C;K;IAGX,qD;MAkBI,WAAO,CAAP,C;QAD2E,  
OAC3D,SAAS,GAAb,GAAkB,GAAIB,GAA2B,MAAM,iBAAiB,GAAjB,EAAcB,KAAtB,EAA6B,IAA7B,CAAN,

I;WACvC,WAAO,CAAP,C;QAF2E,OAE3D,SAAS,GAAb,GAakB,GAAIB,GAA2B,MAAM,iBAAiB,KAAjB,EA  
AwB,GAAxB,EAA6B,CAAC,IAAD,IAA7B,CAAN,I;;QAC/B,MAAA,gCAAyB,eAAzB,C;K;IAGzB,uD;MAkBI,s  
BAAO,CAAP,C;QAD+E,OAC/D,sBAAS,GAAT,MAAJ,GAakB,GAAIB,GAA2B,aAAM,mBAAiB,GAAjB,EAAS  
B,KAAtB,EAA6B,IAA7B,CAAN,C;WACvC,sBAAO,CAAP,C;QAF+E,OAE/D,sBAAS,GAAT,MAAJ,GAakB,G  
AAIB,GAA2B,QAAM,mBAAiB,KAAjB,EAAwB,GAAxB,EAA8B,IAAD,aAA7B,CAAN,C;;QAC/B,MAAA,gCA  
AyB,eAAzB,C;K;IC7DjB,kD;MAAA,8B;MACI,aAAY,C;K;oDACZ,Y;MAAyB,oBAAQ,gBAAI,O;K;iDACrC,Y;  
MAAgD,Q;MAA1B,IAAI,aAAQ,gBAAI,OAAhB,C;QAAA,OAAsB,iBAAI,iBAAJ,EAAl,yBAAJ,O;;QAAkB,MA  
AM,2BAAYB,UAAF,WAAvB,C;K;;IAPhF,oC;MAEI,IAD8D,IAC9D,S;QACI,UAA0B,K;QAF0B,2C;;QAAA,QA  
AM,IAAN,C;eASxD,c;YATwD,OASiC,qBAAqB,KAArB,C;eACIB,W;YAVwD,OAuzC,kBAakB,KAAIB,C;eAC  
f,Y;YAXwD,OAWxC,mBAAMB,KAAAnB,C;eAChB,W;YAZwD,OAYzC,kBAakB,KAAIB,C;eACf,U;YAbwD,O  
Aa1C,iBAAiB,KAAjB,C;eACd,W;YAdwD,OaczC,kBAakB,KAAIB,C;eACf,Y;YafwD,OAexC,mBAAMB,KAA  
nB,C;eAChB,a;YAhBwD,OAgBvC,oBAAoB,KAApB,C;kBACT,MAAM,6BAAsB,2DAA+C,IAA/C,CAAtB,C;;K;  
IAIuC,2D;MAAA,kC;MAAS,0B;MAC9D,aAAY,C;K;2DACZ,Y;MAAyB,oBAAQ,kBAAM,O;K;+DACvC,Y;MA  
A2D,Q;MAA9B,IAAI,aAAQ,kBAAM,OAAIB,C;QAAA,OAAwB,mBAAM,iBAAN,EAAM,yBAAN,O;;QAAoB,  
MAAM,2BAAYB,UAAF,WAAvB,C;K;;IAJnF,qC;MACyD,oD;K;IAON,wD;MAAA,kC;MAAS,uB;MACxD,aAA  
Y,C;K;wDACZ,Y;MAAyB,oBAAQ,kBAAM,O;K;yDACvC,Y;MAAwD,Q;MAA9B,IAAI,aAAQ,kBAAM,OAAIB  
,C;QAAA,OAAwB,mBAAM,iBAAN,EAAM,yBAAN,O;;QAAoB,MAAM,2BAAYB,UAAF,WAAvB,C;K;;IAJhF,  
kC;MACmD,iD;K;IAOE,yD;MAAA,kC;MAAS,wB;MAC1D,aAAY,C;K;yDACZ,Y;MAAyB,oBAAQ,kBAAM,O;  
K;2DACvC,Y;MAAyD,Q;MAA9B,IAAI,aAAQ,kBAAM,OAAIB,C;QAAA,OAAwB,mBAAM,iBAAN,EAAM,yB  
AAN,O;;QAAoB,MAAM,2BAAYB,UAAF,WAAvB,C;K;;IAJf,mC;MACqD,kD;K;IAOF,wD;MAAA,kC;MAAS,  
uB;MACxD,aAAY,C;K;wDACZ,Y;MAAyB,oBAAQ,kBAAM,O;K;yDACvC,Y;MAAwD,Q;MAA9B,IAAI,aAAQ  
,kBAAM,OAAIB,C;QAAA,OAAwB,mBAAM,iBAAN,EAAM,yBAAN,O;;QAAoB,MAAM,2BAAYB,UAAF,WA  
AvB,C;K;;IAJhF,kC;MACmD,iD;K;IAOF,uD;MAAA,kC;MAAS,sB;MACTd,aAAY,C;K;uDACZ,Y;MAAyB,oBA  
AQ,kBAAM,O;K;uDACvC,Y;MAAuD,Q;MAA9B,IAAI,aAAQ,kBAAM,OAAIB,C;QAAA,OAAwB,mBAAM,iB  
AAN,EAAM,yBAAN,O;;QAAoB,MAAM,2BAAYB,UAAF,WAAvB,C;K;;IAJ/E,iC;MACiD,gD;K;IAOI,yD;MAA  
A,kC;MAAS,wB;MAC1D,aAAY,C;K;yDACZ,Y;MAAyB,oBAAQ,kBAAM,O;K;2DACvC,Y;MAAyD,Q;MAA9B  
,IAAI,aAAQ,kBAAM,OAAIB,C;QAAA,OAAwB,mBAAM,iBAAN,EAAM,yBAAN,O;;QAAoB,MAAM,2BAAYB  
,UAAF,WAAvB,C;K;;IAJf,mC;MACqD,kD;K;IAOE,0D;MAAA,kC;MAAS,yB;MAC5D,aAAY,C;K;0DACZ,Y;  
MAAyB,oBAAQ,kBAAM,O;K;6DACvC,Y;MAA0D,Q;MAA9B,IAAI,aAAQ,kBAAM,OAAIB,C;QAAA,OAAwB  
,mBAAM,iBAAN,EAAM,yBAAN,O;;QAAoB,MAAM,2BAAYB,UAAF,WAAvB,C;K;;IAJf,oC;MACuD,mD;K;I  
AOJ,wD;MAAA,kC;MAAS,uB;MACxD,aAAY,C;K;wDACZ,Y;MAAyB,oBAAQ,kBAAM,O;K;yDACvC,Y;MA  
AwD,Q;MAA9B,IAAI,aAAQ,kBAAM,OAAIB,C;QAAA,OAAwB,mBAAM,iBAAN,EAAM,yBAAN,O;;QAAoB,  
MAAM,2BAAYB,UAAF,WAAvB,C;K;;IAJhF,kC;MACmD,iD;K;IAOpB,gC;MAAC,wB;K;;IAEhC,+B;MAC8C,  
MAAM,mC;K;IAEpD,8C;MAEI,IAAI,qBAAJ,C;QACI,OAAO,C5ByIiF,W4BzIrE,U5ByIqE,E4BzIzD,Q5ByIyD,C  
;;Q4BvIxF,OAAS,CAAY,qBAAsB,UAAtB,EAakC,QAAIC,C;;K;IAI7B,2C;MAEI,IAAI,KAAY,kBAAhB,C;QAG  
I,KAAY,mBAakB,QAAIB,C;;QAEH,QAAT,SAA+C,CAAIB,IAAJ,C,KAAiC,EAakB,O;;K;IAIvD,sC;MAGwB,Q;  
MADpB,gBAAgB,IAAhB,KAAgB,E;MACI,IAAI,OCnGkB,ODmGT,OAAT,EAaqB,WAArB,CAAJ,C;QACHB,O  
AAI,aAAJ,GAAmB,KAAM,WAAzB,GAAyC,I;;QAEzC,c;;MAHJ,wB;MAKA,kBAakB,K;MACIB,iBAAiB,W;M  
ACjB,OAAO,S;K;IAIa,sB;MAAC,U;K;iCACrB,iB;MACI,OAAO,mCAAsB,WAAK,KAAM,E;K;mCAG5C,Y;MA  
CI,OAAO,M;K;mCAGX,Y;MACI,OAAuC,oBAAAnB,UAA5B,IAAe,EAAa,CAAM,B,C;K;0CAG3C,iB;MACI,OAA  
R,IAAI,EAaw,GAAN,K;K;kCAGL,Y;MAEI,OAAO,M;K;;+DAIf,gB;MAEI,YAAY,MAAY,IAAK,OAAjB,C;M  
ACZ,sBAAU,IAAV,a;QACI,UAAU,KAak,CAAL,C;QACV,IAAI,oBAAJ,C;UACI,MAAM,CAAN,IAAW,EAAS,  
MAAM,MAAK,GAAL,C;;UAE1B,MAAM,CAAN,IAAW,G;;MAGnB,OAAO,EAAS,OAAO,OAAM,EAAN,EA  
AgB,KAAhB,C;K;IAG3B,2B;MAMW,WAAO,S;MAIbD,YAAY,MAAY,IAAK,OAAjB,C;MACZ,sBAAU,IAAV,  
a;QACI,UAAU,KAak,CAAL,C;QACV,IAAI,oBAAJ,C;UACI,MAAM,CAAN,IAAW,EAAS,MAAM,MAAK,GA  
AL,C;;UAE1B,MAAM,CAAN,IAAW,G;;MAYnB,OATO,EAAS,OAAO,OAAM,EAAN,EAAGB,KAAhB,C;K;IA  
Y3B,oC;MAWI,WAAqB,S;MACrB,IAAI,qBAAM,B,CAAY,OAAd,KAA2B,SAAhD,C;QAjCA,YAAY,MAkCM,I  
AICW,OAAjB,C;QACZ,sBAiCkB,IAjCIB,a;UACI,UAGcC,IAhCJ,CAAK,CAAL,C;UACV,IAAI,oBAAJ,C;YACI,

MAAM,CAAN,IAAW,EAAS,MAAM,MAAK,GAAL,C;;YAE1B,MAAM,CAAN,IAAW,G;;;QA4Bf,OAzBG,EAA  
S,OAAO,OAAM,EAAN,EAAGB,KAAhB,C;;QA2BnB,WAAW,C;QACX,0BAAU,IAAV,e;UACY,IAAoB,I;UAA  
5B,eAAQ,QAAoB,OAAPB,IAAQ,CAAH,GAAG,CAAY,OAAPB,oCAAR,K;;QAEJ,aAAa,IAAjB,CAAC,YAAgB,  
CAAH,IAAG,C;QE3FjB,IF4FyB,CE5FhB,OAAL,KAAkB,SAAtB,C;UF4F4B,ME3FxB,UF2FqB,CE3FF,O;SF4Fn  
B,OAAO,C;QACP,0BAAU,IAAV,e;UAE0B,YACX,M;UAFX,YAAU,IAAQ,CAAH,GAAG,C;UACI,SAAJ,KAAI  
,O;UAAtB,aAAU,CAAV,kB;YACI,OAAO,aAAP,EAAO,qBAAP,YAAiB,MAAI,CAAJ,C;;QAGzB,OAAO,M;;K;  
IAIf,0B;MACgC,WAAS,c;MAAT,YAAhC,EAAE,MAAM,KAAiD,CAA3C,SAA2C,C;MAWrD,eAAiB,I;MAXW,  
OAYrB,K;K;IAVX,uB;MAC6B,WAAS,W;MAAT,YAAsB,IAA/C,WAA+C,CAAnC,EAAE,MAAM,KAAK,CAA  
C,SAAD,CAAsB,C;MAQ/C,eAAiB,I;MARQ,OASiB,K;K;IAPX,uB;MAC6B,WAAS,W;MAAT,YAA7B,EAAE,M  
AAM,KAA2C,CAArC,SAAqC,C;MAK/C,eAAiB,I;MALQ,OAMiB,K;K;2DAJX,uB;MAGI,eAAiB,I;MACjB,OA  
AO,K;K;KEG9MX,yB;MAAA,0B;MAAA,uB;QASI,OAAoB,OAAb,ItD0Q+B,KAAAL,GAAiB,KsD1Q9B,C;O;KA  
TxB,C;ICIqC,2C;MAAC,8C;MACiC,eAAsB,C;MACtB,wBAA+B,C;MAC/B,gBAA6B,I;MAC7B,mBAAsC,I;MA  
CtC,qBAAYC,I;MAEzC,yBAAGD,yBAAMB,Q;MAEnE,sBAAGD,I;K;wFAFhD,Y;MAAA,6B;K;OCAIA,Y;MAEY  
,kBADR,M;MAAA,U;MAAA,2C;QAAA,e;;QAES,gBADD,2CAAQ,yCAAR,gDAAwD,IAAxD,6BAAiE,I;QACz  
D,sB1CwEd,S;Q0C1EF,S1C2EG,S;;M0C3EH,a;K;iDAIJ,kB;MACI,kBAAC,IAAd,C;MACiC,oB;MCuBrB,Q;MAD  
R,IDtBsB,MCsBtB,W;QADJ,mBACiB,I;;QADjB,mBAEY,QDvBc,MCuBd,+D;;MDvBZ,yC;MACA,2BAAMC,M  
AAO,kBAA1C,C;MAGA,OAAO,IAAP,C;Q1CoCY,gB0CnCH,S;;QACD,iBAAiB,8B;QAGjB,IAAI,0BAAJ,C;UA  
CI,qBAAC,e;;UAEd,oBAAQ,0B;UACR,wBAAy,kB;;;UAIZ,cAAc,oB;UACd,IAAI,YAAY,yBAAhB,C;YAAqC,M  
;UACrC,kBAAGB,O;UACHB,qBAAMB,I;;UAEnB,kBAAGB,I;UACHB,qBAAMB,S;;QAGvB,gC;QAEA,IAAI,wC  
AAJ,C;UAEI,YAAU,U;;UAGV,U;UAAA,0C;YETHB,8BDgDQ,WAAO,qBAAP,CChDR,C;YFSgB,a;;YAAA,a;U  
AAA,mB;YAEK,UEpBrB,oBDgDQ,WD5B+B,eC4B/B,CChDR,C;WFqBgB,M;;;K;mDAMhB,Y;MACI,kBAAkB,  
mB;MACIB,IAAI,uBAAuB,gBAAGB,IAA3C,C;QACI,uCAAQ,yCAAR,EAAmC,wCAA+B,WAA/B,C;OAEvC,sB  
AAoB,mC;K;;IAM5B,iC;MAAA,qC;K;gGAEQ,Y;M7C0DyC,MAAM,6B6C1DjC,uC7C0D+D,WAA9B,C;K;yD6  
CxDnD,kB;M7CwD6C,MAAM,6B6CvDzC,uC7CuDuE,WAA9B,C;K;+C6CpDnD,Y;MAAkC,8C;K;;IARtC,6C;  
MAAA,4C;QAAA,2B;OAAA,qC;K;IGyDA,mG;IAAA,yH;IAAA,6F;MAKW,kC;MAAS,4C;K;IALpB,sEAMQ,Y;  
MACI,Q;MAAA,sC;QAAiB,U;OACjB,OAAO,oB;K;IARnB,6G;sJAjIA,iC;MAgBU,OAAK,SAAL,CAAiB,UAAj  
B,EAA6B,KAA7B,C;K;wJAEV,2C;MAiBU,OAAK,SAAL,CAAiB,QAAjB,EAA2B,UAA3B,EAAuC,KAAvC,C;K  
;wJAEV,kD;MAKU,OAAK,SAAL,CAAiB,QAAjB,EAA2B,KAA3B,EAAkC,UAAIC,EAA8C,KAA9C,C;K;IAGC6  
C,oG;MAAA,mB;QAC3C,OAAK,iCAAL,CAAiB,kBAAjB,C;O;K;IA/BZ,6D;MA0BI,IAAS,SAAY,OAAjB,IAA2  
B,CAA/B,C;QAAA,OAES,SAAL,CAAiB,UAAjB,EAA6B,IAA7B,C;;QA8D0B,Q;QAhE9B,4DAImD,0DAJnD,E  
AgE8B,qBA5DS,UA4DT,qCAhE9B,C;;K;IAwCmD,wH;MAAA,mB;QAC3C,OAAK,iCAAL,CAAiB,gBAAjB,EA  
A2B,kBAA3B,C;O;K;IAhCZ,yE;MA2BI,IAAS,SAAY,OAAjB,IAA2B,CAA/B,C;QAAA,OAES,SAAL,CAAiB,Q  
AAjB,EAA2B,UAA3B,EAAuC,IAAvC,C;;QA0B0B,Q;QA5B9B,4DAImD,sEAJnD,EA4B8B,qBAxBS,UAWBT,q  
CA5B9B,C;;K;IASJ,gC;MAWK,kBAAD,M;MAAA,kBAAC,qEAAD,4DAA2C,S;K;6CAG/C,yB;MAAA,mG;MA  
AA,yH;MAAA,6F;QAKW,kC;QAAS,4C;O;MALpB,sEAMQ,Y;QACI,Q;QAAA,sC;UAAiB,U;SACjB,OAAO,oB;  
O;MARnB,6G;MAAA,oC;QAKkC,Q;QAA9B,mEAA8B,oEAA9B,C;O;KALJ,C;IFC7HA,a;MAC6C,OAAA,MAA  
a,YAAW,CAAX,C;K;ICM3B,iC;;MAA6E,Q;MAAA,+BAAS,I;sCAAIB,O,2DAAA,O;;;K;;;;;IAC/F,2B;MAAA,  
iD;MAAuB,oBAAK,IAAL,EAAW,IAAX,C;MAAvB,Y;K;IACA,sC;MAAA,iD;MAAuC,oBAAK,OAAL,EAAC,IA  
Ad,C;MAAvC,Y;K;IACA,oC;MAAA,iD;MAAwC,oBAAK,SAAL,EAAGB,KAAhB,C;MAAxC,Y;K;IAI+B,mC;;  
MAA6E,Q;MAAA,+BAAS,I;sCAAIB,O,2DAAA,O;;;K;;;;;IACnG,+B;MAAA,mD;MAAuB,sBAAK,IAAL,EA  
AW,IAAX,C;MAAvB,Y;K;IACA,0C;MAAA,mD;MAAuC,sBAAK,OAAL,EAAC,IAAd,C;MAAvC,Y;K;IACA,w  
C;MAAA,mD;MAAwC,sBAAK,SAAL,EAAGB,KAAhB,C;MAAxC,Y;K;IAGsC,0C;MAA0D,qBAAU,OAAV,EA  
AmB,KAAAnB,C;;K;;IACHG,sC;MAAA,0D;MAAuB,6BAAK,IAAL,EAAW,IAAX,C;MAAvB,Y;K;IACA,iD;MA  
AA,0D;MAAuC,6BAAK,OAAL,EAAC,IAAd,C;MAAvC,Y;K;IACA,+C;MAAA,0D;MAAwC,6BAAK,SAAL,EA  
AGB,KAAhB,C;MAAxC,Y;K;IAG8C,kD;MAA0D,4BAAiB,OAAjB,EAA0B,KAA1B,C;;K;;IACxG,8C;MAAA,kE  
;MAAuB,qCAAK,IAAL,EAAW,IAAX,C;MAAvB,Y;K;IACA,yD;MAAA,kE;MAAuC,qCAAK,OAAL,EAAC,IAA  
d,C;MAAvC,Y;K;IACA,uD;MAAA,kE;MAAwC,qCAAK,SAAL,EAAGB,KAAhB,C;MAAxC,Y;K;IAG2C,+C;M  
AA0D,4BAAiB,OAAjB,EAA0B,KAA1B,C;;K;;IACrG,2C;MAAA,+D;MAAuB,kCAAK,IAAL,EAAW,IAAX,C;M

AAvB,Y;K;IACA,sD;MAAA,+D;MAAuC,kCAAK,OAAL,EAAC,IAAd,C;MAAvC,Y;K;IACA,oD;MAAA,+D;M  
AAwC,kCAAK,SAAL,EAAGB,KAAhB,C;MAAxC,Y;K;IAG+C,4C;8BAAwD,O;;K;;IACvG,+C;MAAA,mE;MA  
AuB,sCAAK,IAAL,C;MAAvB,Y;K;IAGqD,yD;MAA0D,4BAAiB,OAAjB,EAA0B,KAA1B,C;;K;;IAC/G,qD;MA  
AA,yE;MAAuB,4CAAK,IAAL,EAAW,IAAX,C;MAAvB,Y;K;IACA,gE;MAAA,yE;MAAuC,4CAAK,OAAL,EA  
Ac,IAAd,C;MAAvC,Y;K;IACA,8D;MAAA,yE;MAAwC,4CAAK,SAAL,EAAGB,KAAhB,C;MAAxC,Y;K;IAGm  
D,uD;MAA0D,4BAAiB,OAAjB,EAA0B,KAA1B,C;;K;;IAC7G,mD;MAAA,uE;MAAuB,0CAAK,IAAL,EAAW,I  
AAX,C;MAAvB,Y;K;IACA,8D;MAAA,uE;MAAuC,0CAAK,OAAL,EAAC,IAAd,C;MAAvC,Y;K;IACA,4D;MA  
AA,uE;MAAwC,0CAAK,SAAL,EAAGB,KAAhB,C;MAAxC,Y;K;IAI2C,wC;sCAAG,E,O;;K;;IAC3G,2C;MAAA,+  
D;MAAuB,kCAAK,IAAL,C;MAAvB,Y;K;IAI0C,uC;8BAAwD,O;;K;;IACIG,0C;MAAA,8D;MAAuB,iCAAK,IA  
AL,C;MAAvB,Y;K;IAGwC,qC;8BAAwD,O;;K;;IACHG,wC;MAAA,4D;MAAuB,+BAAK,IAAL,C;MAAvB,Y;K;I  
AIJ,wC;MACmD,mBAAM,OAAN,EAAC,KAAf,C;;K;;IAC/C,oC;MAAA,wD;MAAuB,sBAAK,IAAL,Q;MAAvB,  
Y;K;IACA,+C;MAAA,wD;MAAGC,2BAAK,OAAL,EAAC,IAAd,C;MAAhC,Y;K;IACA,+C;MAAA,wD;MAAiD,I  
AAY,I;MAAZB,2BAAa,SAAR,OAAQ,CAAb,EAAYB,sDAAzB,C;MAApC,Y;K;IAG4C,yC;8BAAwD,O;;K;;IACp  
G,4C;MAAA,gE;MAAuB,mCAAK,IAAL,C;MAAvB,Y;K;IAIyC,sC;8BAAwD,O;;K;;IACjG,yC;MAAA,6D;MAA  
uB,gCAAK,IAAL,C;MAAvB,Y;K;IAGkD,sD;MAA0D,4BAAiB,OAAjB,EAA0B,KAA1B,C;;K;;IAC5G,kD;MAA  
A,sE;MAAuB,yCAAK,IAAL,EAAW,IAAX,C;MAAvB,Y;K;IACA,6D;MAAA,sE;MAAuC,yCAAK,OAAL,EAAC  
,IAAd,C;MAAvC,Y;K;IACA,2D;MAAA,sE;MAAwC,yCAAK,SAAL,EAAGB,KAAhB,C;MAAxC,Y;K;IAG0D,8  
D;MAA0D,4BAAiB,OAAjB,EAA0B,KAA1B,C;;K;;IACpH,0D;MAAA,8E;MAAuB,iDAAK,IAAL,EAAW,IAAX,  
C;MAAvB,Y;K;IACA,qE;MAAA,8E;MAAuC,iDAAK,OAAL,EAAC,IAAd,C;MAAvC,Y;K;IACA,mE;MAAA,8E;  
MAAwC,iDAAK,SAAL,EAAGB,KAAhB,C;MAAxC,Y;K;6FCIGJ,yB;MAEI,OAAG,GAAG,CAAC,QAAD,C;K;m  
FAGV,oB;MAEI,OAAG,GAAG,GAAG,G;K;6ETVN,a;MAK8C,cAAvC,C;K;6EHP,Y;MAG+C,S;K;IA6B/C,2B;M  
AG4D,0BAAe,WAAf,C;K;IAE5D,mC;MAIwF,0BAAe,WAAf,C;K;IAExF,mC;MAKwE,0BAAe,WAAf,C;K;IAG  
xE,4B;MAI8D,Q;MAH1D,aAAkB,GAAL,O;MACTb,aAAkB,GAAL,O;MACTb,YAAiB,C;MACjB,OOAO,QAAQ,  
MAAR,IAAkB,QAAQ,MAAjC,C;QAAyC,IAAI,KAAJ,IAAa,IAAI,YAAJ,EAAL,oBAAJ,O;;MACTd,OOAO,G;K;I  
AIX,wD;MAMuC,Q;MALnC,aAAa,MAAO,OAAM,CAAN,EAAS,OAAT,C;MA0BpB,IAzBc,MAyBL,OAAL,KA  
AkB,SAAtB,C;QAZbSb,MA0BIB,UA1BU,MA0BS,O;OAZbVb,YAAiB,MAAO,O;MACxB,IAAI,UAAU,KAAAd,C  
;QACI,gBAAGB,O;QACHB,OOAO,QAAQ,OAAf,C;UAAwB,OOAO,YAAP,EAAL,oBAAP,UAAkB,Y;;OAE9C,  
OOAO,M;K;IAGX,gD;MAKoB,UAAmB,M;MAJnC,aAAa,KAAM,Q;MACnB,MAAO,OAAP,IAAiB,UAAW,K;  
MAC5B,IAbc,KAAI,OAAL,KAkB,SAAtB,C;QAbqB,MACjB,UAdU,KAcS,O;OAbvB,YAAiB,KAAM,O;MACP,  
4B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAY,OOAO,cAAP,EAAL,sBAAP,YAAkB,O;;MAC9C,OOAO,M;  
K;IAGX,yD;MAEOB,UAAgB,M;MADhC,YAAY,U;MACI,4B;MAAhB,OAAGB,cAAhB,C;QAAGB,yB;QAAY,IA  
AI,cAAJ,EAAL,sBAAJ,YAAe,O;;MAC3C,OOAO,G;K;oFAGX,oB;MACI,IAAI,IAAK,OAAL,KAkB,SAAtB,C;  
QACI,YAAc,IAAK,O;Q;0EAI3B,wB;MAA+D,OOAA,MAAa,QAAO,GAAP,EAAY,OOAZ,C;K;IS/F5E,mC;MA  
OI,kBAkB,MAAa,eAAc,SAAd,C;MAC/B,iBAAiB,MAAa,eAAc,IAAd,C;MAC9B,OOAW,gBAAe,UAAAnB,GA  
A+B,SAA/B,GAAYC,CAAC,S;K;0ECUrD,2B;MAKyE,OOAA,MAAa,gBAAe,IAAf,C;K;4EAYBtF,2B;MAKsE,O  
AAA,MAAa,eAAc,IAAd,C;K;kEAGnF,qB;MACgD,OOAA,MAAa,KAAC,UAAS,GAAT,EAAC,IAAd,C;K;wEAC  
hC,qB;MAAQ,OAAC,SAAY,a;K;0EACxB,qB;MAAQ,OAAC,SAAY,c;K;IC3D5D,0D;MAGI,OOAO,I;K;ICHX,s  
C;MAMsD,OOAA,SAAY,UAAS,WAAW,KAAX,CAAT,C;K;ItDKIE,uC;Mf2nBW,Q;MAAA,IernBgB,KfqBZ,I  
AAS,CAAT,IernBY,KfqBZ,IAAS,wBAA3B,C;QAAA,OOAsC,UernBtB,KfqBZ,C;;QernBb,MAAM,8BAA0B,i  
CAAuB,gBAAvB,MAA1B,C;;MAAtC,W;K;IAGJ,uC;Mf4nBW,Q;MAAA,IetnBgB,KfsnBZ,IAAS,CAAT,IetnBY,  
KfsnBZ,IAAS,0BAA3B,C;QAAA,OOAsC,UetnBtB,KfsnBZ,C;;QetnBb,MAAM,8BAA0B,iCAAuB,gBAAvB,M  
AA1B,C;;MAAtC,W;K;IAGJ,uC;Mf6nBW,Q;MAAA,IevnBgB,KfunBZ,IAAS,CAAT,IevnBY,KfunBZ,IAAS,0B  
AA3B,C;QAAA,OOAsC,UevnBtB,KfunBZ,C;;QevnBb,MAAM,8BAA0B,iCAAuB,gBAAvB,MAA1B,C;;MAAtC,  
W;K;IAGJ,uC;Mf8nBW,Q;MAAA,IexnBgB,KfwnBZ,IAAS,CAAT,IexnBY,KfwnBZ,IAAS,0BAA3B,C;QAAA,O  
OAsC,UexnBtB,KfwnBZ,C;;QexnBb,MAAM,8BAA0B,iCAAuB,gBAAvB,MAA1B,C;;MAAtC,W;K;IAGJ,uC;M  
f+nBW,Q;MAAA,IeznBgB,KfynBZ,IAAS,CAAT,IeznBY,KfynBZ,IAAS,0BAA3B,C;QAAA,OOAsC,UeznBtB,Kf  
ynBZ,C;;QeznBb,MAAM,8BAA0B,iCAAuB,gBAAvB,MAA1B,C;;MAAtC,W;K;IAGJ,uC;MfgoBW,Q;MAAA,I  
e1nBgB,Kf0nBZ,IAAS,CAAT,Ie1nBY,Kf0nBZ,IAAS,0BAA3B,C;QAAA,OOAsC,Ue1nBtB,Kf0nBZ,C;;Qe1nBb,

MAAM,8BAA0B,iCAAuB,gBAAvB,MAA1B,C,;MAAtC,W;K;IAGJ,uC;MfioBW,Q;MAAA,Ie3nBgB,Kf2nBZ,IAAS,CAAT,Ie3nBY,Kf2nBE,IAAS,0BAA3B,C;QAAA,OAAcC,Ue3nBtB,Kf2nBsB,C,;Qe3nBb,MAAM,8BAA0B,iCAAuB,gBAAvB,MAA1B,C,;MAAtC,W;K;IAGJ,uC;MfkoBW,Q;MAAA,Ie5nBgB,Kf4nBZ,IAAS,CAAT,Ie5nBY,Kf4nBE,IAAS,0BAA3B,C;QAAA,OAAcC,Ue5nBtB,Kf4nBsB,C,;Qe5nBb,MAAM,8BAA0B,iCAAuB,gBAAvB,MAA1B,C,;MAAtC,W;K;IAGJ,wC;MfmoBW,Q;MAAA,Ie7nBgB,Kf6nBZ,IAAS,CAAT,Ie7nBY,Kf6nBE,IAAS,0BAA3B,C;QAAA,OAAcC,Ue7nBtB,Kf6nBsB,C,;Qe7nBb,MAAM,8BAA0B,iCAAuB,gBAAvB,MAA1B,C,;MAAtC,W;K;IAGJ,2B;MAII,OAAO,cAAa,SAAb,C;K;oFAGX,yB;MAAA,gD;MAAA,4B;QAKI,OAAcC,OAA/B,SAA+B,C;O;KAL1C,C;oFAQA,yB;MAAA,gD;MAAA,4B;QAKI,OAAuC,OAAhC,SAAgC,C;O;KAL3C,C;oFAQA,yB;MAAA,gD;MAAA,4B;QAKI,OAAqC,OAA9B,SAA8B,C;O;KALzC,C;oFAQA,yB;MAAA,gD;MAAA,4B;QAKI,OAAcC,OAA/B,SAA+B,C;O;KAL1C,C;oFAQA,yB;MAAA,gD;MAAA,4B;QAKI,OAAuC,OAAhC,SAAgC,C;O;KAL3C,C;oFAQA,yB;MAAA,gD;MAAA,4B;QAKI,OAAwC,OAAjC,SAAiC,C;O;KAL5C,C;oFAQA,yB;MAAA,gD;MAAA,4B;QAKI,OAAyC,OAAIC,SAAkC,C;O;KAL7C,C;IAYW,2C;MAAA,8B;MAAS,uB;K;4FACW,Y;MAAQ,OAAA,gBAAY,O;K;6CAC3C,Y;MAAkC,OAAA,gBfunP/B,YAAQ,C;K;oDetnPX,mB;MAAgD,OAAy,WAAZ,gBAAY,EAAS,OAAAT,C;K;iDAC5D,iB;MACI,oCAAa,2BAAkB,KAAIB,EAAyB,SAAzB,C;MACb,OAAO,6BAAY,KAAZ,E;K;mDAEX,mB;MAES,Q;MAAL,IAAI,eAAC,uFAAD,CAAJ,C;QAAgC,OAAO,E;MACvC,OAAmB,UAAZ,gBAAY,EAAQ,OAAAR,C;K;uDAEvB,mB;MAES,Q;MAAL,IAAI,eAAC,uFAAD,CAAJ,C;QAAgC,OAAO,E;MACvC,OAAmB,cAAZ,gBAAY,EAAy,OAAZ,C;K;IApB/B,6B;MAII,0C;K;IAqBJ,+C;MAaI,OAAy,kBAAL,SAAK,EAAkB,KAAIB,C;K;IAqBhB,0C;MASI,OAAy,oBAAL,SAAK,C;K;IAehB,0C;MAYI,OAAy,oBAAL,SAAK,C;K;IAkBhB,2C;MAWI,OAAy,cAAL,SAAK,EAAc,KAAAd,C;K;IAGhB,2C;MAWI,OAAy,cAAL,SAAK,EAAc,KAAAd,C;K;IAGhB,4C;MAWI,OAAy,cAAL,SAAK,EAAc,KAAAd,C;K;IAGhB,4C;MAWI,OAAy,cAAL,SAAK,EAAc,KAAAd,C;K;IAGhB,4C;MAWI,OAAy,cAAL,SAAK,EAAc,KAAAd,C;K;IAGhB,4C;MAWI,OAAy,cAAL,SAAK,EAAc,KAAAd,C;K;IAGhB,4C;MAWI,OAAy,cAAL,SAAK,EAAc,KAAAd,C;K;IAwHhB,sC;MAOI,OAAy,gBAAL,SAAK,C;K;IAGhB,sC;MAOI,OAAy,gBAAL,SAAK,C;K;IAGhB,uC;MAOI,OAAy,gBAAL,SAAK,C;K;IAGhB,uC;MAOI,OAAy,gBAAL,SAAK,C;K;IAGhB,uC;MAOI,OAAy,gBAAL,SAAK,C;K;IAGhB,uC;MAOI,OAAy,gBAAL,SAAK,C;K;IAoFhB,sC;MASI,OAAy,gBAAL,SAAK,C;K;IAGhB,sC;MASI,OAAy,gBAAL,SAAK,C;K;IAGhB,uC;MASI,OAAy,gBAAL,SAAK,C;K;IAGhB,uC;MASI,OAAy,gBAAL,SAAK,C;K;IAGhB,uC;MASI,OAAy,gBAAL,SAAK,C;K;IAGhB,uC;MASI,OAAy,gBAAL,SAAK,C;K;IAGhB,uC;MASI,OAAy,gBAAL,SAAK,C;K;wFAsGhB,yB;MAAA,8C;MAAA,kF;QAmB0E,iC;UAAA,oBAAYB,C;QAAG,0B;UAAA,aAAkB,C;QAAG,wB;UAAA,WAAgB,gB;QACvI,UAAU,SAAV,EAAgB,WAAhB,EAA6B,iBAA7B,EAAgD,UAAhD,EAA4D,QAA5D,C;QACA,OAAO,W;O;KArBX,C;wFAwBA,yB;MAAA,8C;MAAA,kF;QAmBoE,iC;UAAA,oBAAYB,C;QAAG,0B;UAAA,aAAkB,C;QAAG,wB;UAAA,WAAgB,gB;QACjI,UAAU,SAAV,EAA0C,WAA1C,EAAiF,iBAAjF,EAAoG,UAApG,EAAgH,QAAhH,C;QACA,OAAO,W;O;KArBX,C;wFAwBA,yB;MAAA,8C;MAAA,kF;QAmBsE,iC;UAAA,oBAAYB,C;QAAG,0B;UAAA,aAAkB,C;QAAG,wB;UAAA,WAAgB,gB;QACnI,UAAU,SAAV,EAA2C,WAA3C,EAAmF,iBAAnF,EAAgG,UAAATG,EAAkH,QAAIH,C;QACA,OAAO,W;O;KArBX,C;wFAwBA,yB;MAAA,8C;MAAA,kF;QAmBkE,iC;UAAA,oBAAYB,C;QAAG,0B;UAAA,aAAkB,C;QAAG,wB;UAAA,WAAgB,gB;QAC/H,UAAU,SAAV,EAAyC,WAAzC,EAA+E,iBAA/E,EAAkG,UAAIG,EAA8G,QAA9G,C;QACA,OAAO,W;O;KArBX,C;wFAwBA,yB;MAAA,8C;MAAA,kF;QAmBoE,iC;UAAA,oBAAYB,C;QAAG,0B;UAAA,aAAkB,C;QAAG,wB;UAAA,WAAgB,gB;QACrI,UAAU,SAAV,EAA4C,WAA5C,EAAqF,iBAArF,EAAwG,UAAxG,EAAoH,QAApH,C;QACA,OAAO,W;O;KArBX,C;wFAwBA,yB;MAAA,8C;MAAA,kF;QAmB0E,iC;UAAA,oBAAYB,C;QAAG,0B;UAAA,aAAkB,C;QAAG,wB;UAAA,WAAgB,gB;QACvI,UAAU,SAAV,EAA6C,WAA7C,EAAuF,iBAAvF,EAA0G,UAA1G,EAAsh,QAAtH

,C;QACA,OAAO,W;O;KArBX,C;yFAwBA,yB;MAAA,8C;MAAA,kF;QAmBoE,iC;UAAA,oBAAyB,C;QAAG,0  
B;UAAA,aAAkB,C;QAAG,wB;UAAA,WAAGB,gB;QACjI,UAAU,SAAV,EAA0C,WAA1C,EAAiF,iBAAjF,EAA  
oG,UAApG,EAAgH,QAaHh,C;QACA,OAAO,W;O;KArBX,C;oFAwBA,qB;MAOI,OAAY,SAAY,Q;K;oFAG5B,  
qB;MAOI,OAAY,SAAY,Q;K;oFAG5B,qB;MAOI,OAAY,SAAY,Q;K;qFAG5B,qB;MAOI,OAAY,SAAY,Q;K;IA  
G5B,8B;MAMW,WAAS,W;MAAT,YAA2B,SAAY,Q;MwC17B9C,eAAiB,I;MxCk7BjB,OwCj7BO,K;K;qFxC07B  
X,qB;MAOI,OAAY,SAAY,Q;K;qFAG5B,qB;MAOI,OAAY,SAAY,Q;K;IAG5B,8B;MAMW,WAAS,c;MAAT,YA  
A8B,SAAY,Q;MwC/8BjD,eAAiB,I;MxC+8BjB,OwC98BO,K;K;IxCi9BX,8B;MAMW,WAAS,W;MAAT,YAA2B,  
SAAY,Q;MwCx9B9C,eAAiB,I;MxCw9BjB,OwCv9BO,K;K;IxCO9BX,uC;MD5oCI,IAAI,ECspCI,WAAW,CDtpC  
f,CAAJ,C;QACI,cCqpCoB,0C;QDppCpB,MAAM,gCAAYB,OAAQ,WAAjC,C;OCqpCV,OAAO,SAAS,SAAT,EA  
Ae,cAAU,OAAV,CAAf,C;K;IAGX,uC;MD1pCI,IAAI,ECqCI,WAAW,CDpqCf,CAAJ,C;QACI,cCmqCoB,0C;Q  
DlqCpB,MAAM,gCAAYB,OAAQ,WAAjC,C;OCmqCV,OAAO,SAAS,SAAT,EAAe,eAAW,OAAX,CAAf,C;K;IA  
GX,uC;MDxqCI,IAAI,ECkrCI,WAAW,CDlrCf,CAAJ,C;QACI,cCirCoB,0C;QDhrCpB,MAAM,gCAAYB,OAAQ,  
WAAjC,C;OCirCV,OAAO,SAAS,SAAT,EAAe,eAAS,OAAT,CAAf,C;K;IAGX,uC;MDtrCI,IAAI,ECgsCI,WAA  
W,CDhsCf,CAAJ,C;QACI,cC+rCoB,0C;QD9rCpB,MAAM,gCAAYB,OAAQ,WAAjC,C;OC+rCH,WAAS,W;MA  
AT,YAAsB,gBAAGB,SAAhB,EAA8B,OAAtB,K;MwChhC7B,eAAiB,I;MxCghCjB,OwC/gCO,K;K;IxCkhCX,uC;  
MDpsCI,IAAI,EC8sCI,WAAW,CD9sCf,CAAJ,C;QACI,cC6sCoB,0C;QD5sCpB,MAAM,gCAAYB,OAAQ,WAAj  
C,C;OC6sCV,OAAO,SAAS,SAAT,EAAe,iBAAW,OAAX,CAAf,C;K;IAGX,uC;MDltCI,IAAI,EC4tCI,WAAW,C  
D5tCf,CAAJ,C;QACI,cC2tCoB,0C;QD1tCpB,MAAM,gCAAYB,OAAQ,WAAjC,C;OC2tCV,OAAO,SAAS,SAAT,  
EAAe,iBAAY,OAAs,CAAf,C;K;IAGX,uC;MDhuCI,IAAI,EC0uCI,WAAW,CD1uCf,CAAJ,C;QACI,cCyCoB,0C  
;QDxuCpB,MAAM,gCAAYB,OAAQ,WAAjC,C;OCyuCH,WAAS,c;MAAT,YAAyB,gBAAGB,SAAhB,EAA8B,O  
AAtB,EAA+B,KAA/B,C;MwC1jChC,eAAiB,I;MxC0jCjB,OwCzjCO,K;K;IxC4jCX,uC;MD9uCI,IAAI,ECwvCI,W  
AAW,CDxvCf,CAAJ,C;QACI,cCuvCoB,0C;QDtvCpB,MAAM,gCAAYB,OAAQ,WAAjC,C;OCuvCH,WAAS,W;  
MAAT,YAAsB,SAAS,SAAT,EAAe,iBAAU,OAAV,CAAf,C;MwCxc7B,eAAiB,I;MxCwkCjB,OwCvkCO,K;K;I  
xC0kCX,uC;MD5vCI,IAAI,ECuwCI,WAAW,CDvwCf,CAAJ,C;QACI,cCswCoB,0C;QDrwCpB,MAAM,gCAAYB  
,OAAQ,WAAjC,C;OCswCV,OAAO,gBAAGB,SAAhB,EAA8B,OAAtB,EAA+B,IAA/B,C;K;IAGX,sD;MAWI,oC  
AAa,2BAAkB,SAIb,EAA6B,OAA7B,EAA8B,gBAAtC,C;MACb,OAAY,SAAY,OAAM,SAAN,EAAiB,OAAjB,  
C;K;IAG5B,sD;MAUI,oCAAA,2BAAkB,SAIb,EAA6B,OAA7B,EAA8B,gBAAtC,C;MACb,OAAY,SAAY,OAAM,  
SAAN,EAAiB,OAAjB,C;K;IAG5B,sD;MAUI,oCAAA,2BAAkB,SAIb,EAA6B,OAA7B,EAA8B,gBAAtC,C;M  
ACb,OAAY,SAAY,OAAM,SAAN,EAAiB,OAAjB,C;K;IAG5B,sD;MAUI,oCAAA,2BAAkB,SAIb,EAA6B,OAA  
7B,EAA8B,gBAAtC,C;MACb,OAAY,SAAY,OAAM,SAAN,EAAiB,OAAjB,C;K;IAG5B,sD;MAUI,oCAAA,2BA  
AkB,SAIb,EAA6B,OAA7B,EAA8B,gBAAtC,C;MACN,WAAS,W;MAAT,YAA2B,SAAY,OAAM,SAAN,EAAi  
B,OAAjB,C;MwC9pC9C,eAAiB,I;MxC8pCjB,OwC7pCO,K;K;IxCgqCX,sD;MAUI,oCAAA,2BAAkB,SAIb,EA  
A6B,OAA7B,EAA8B,gBAAtC,C;MACb,OAAY,SAAY,OAAM,SAAN,EAAiB,OAAjB,C;K;IAG5B,sD;MAUI,oC  
AAa,2BAAkB,SAIb,EAA6B,OAA7B,EAA8B,gBAAtC,C;MACb,OAAY,SAAY,OAAM,SAAN,EAAiB,OAAjB,  
C;K;IAG5B,uD;MAUI,oCAAA,2BAAkB,SAIb,EAA6B,OAA7B,EAA8B,gBAAtC,C;MACN,WAAS,c;MAAT,Y  
AA8B,SAAY,OAAM,SAAN,EAAiB,OAAjB,C;MwCxsCjD,eAAiB,I;MxCwsCjB,OwCvsCO,K;K;IxCOsCX,uD;M  
AUI,oCAAA,2BAAkB,SAIb,EAA6B,OAA7B,EAA8B,gBAAtC,C;MACN,WAAS,W;MAAT,YAA2B,SAAY,OA  
AM,SAAN,EAAiB,OAAjB,C;MwCttC9C,eAAiB,I;MxCstCjB,OwCrtCO,K;K;IxCwtCX,wD;MAWgD,yB;QAAA,  
YAAiB,C;MAAG,uB;QAAA,UAAe,gB;MAC/E,oCAAA,2BAAkB,SAIb,EAA6B,OAA7B,EAA8B,gBAAtC,C;M  
ACR,SAAY,MAAK,OAAL,EAAC,SAAd,EAAYB,OAazB,C;K;IAGrB,wD;MAWgD,yB;QAAA,YAAiB,C;MAAG  
,uB;QAAA,UAAe,gB;MAC/E,oCAAA,2BAAkB,SAIb,EAA6B,OAA7B,EAA8B,gBAAtC,C;MACR,SAAY,MAA  
K,OAAL,EAAC,SAAd,EAAYB,OAazB,C;K;IAGrB,wD;MAWkD,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe  
,gB;MACjF,oCAAA,2BAAkB,SAIb,EAA6B,OAA7B,EAA8B,gBAAtC,C;MACR,SAAY,MAAK,OAAL,EAAC,S  
AAd,EAAYB,OAazB,C;K;IAGrB,wD;MAW8C,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,gB;MAC7E,oCA  
AA,2BAAkB,SAIb,EAA6B,OAA7B,EAA8B,gBAAtC,C;MACR,SAAY,MAAK,OAAL,EAAC,SAAd,EAAYB,OA  
AzB,C;K;IAGrB,wD;MAWgD,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,gB;MAC/E,oCAAA,2BAAkB,SAI  
b,EAA6B,OAA7B,EAA8B,gBAAtC,C;MACR,SAAY,MAAK,OAAL,EAAC,SAAd,EAAYB,OAazB,C;K;IAGrB,  
wD;MAWkD,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,gB;MACjF,oCAAA,2BAAkB,SAIb,EAA6B,OAA

7B,EAAsC,gBAAtC,C;MACR,SAAY,MAAK,OAAL,EAAC,SAAd,EAAYB,OAAzB,C;K;IAGrB,wD;MAWoD,yB; QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,gB;MACnF,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,gBAA tC,C;MACR,SAAY,MAAK,OAAL,EAAC,SAAd,EAAYB,OAAzB,C;K;IAGrB,yD;MAWsD,yB;QAAA,YAAiB,C; MAAG,uB;QAAA,UAAe,gB;MACrF,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,gBAAtC,C;MACR,SAAY,MAAK,OAAL,EAAC,SAAd,EAAYB,OAAzB,C;K;IAGrB,yD;MAWgD,yB;QAAA,YAAiB,C;MAAG,uB;QAA A,UAAe,gB;MAC/E,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,gBAAtC,C;MACR,SAAY,MAAK,OAAL ,EAAC,SAAd,EAAYB,OAAzB,C;K;iFAGrB,8B;MAKI,OAAy,SAAY,QAAO,CAAQ,OAAR,CAAP,C;K;iFAG5B, yB;MAwIA,iD;MAxIA,qC;QAKI,OAwoIO,gCAxIK,eAAY,OAAZ,EAwIL,C;O;KA7IX,C;iFAQA,yB;MAwIA,iD; MAxIA,qC;QAKI,OAwoIO,gCAxIK,gBAAa,OAAb,EAwIL,C;O;KA7IX,C;iFAQA,yB;MAwIA,iD;MAxIA,qC;QA KI,OAwoIO,gCAxIK,gBAAW,OAAx,EAwIL,C;O;KA7IX,C;iFAQA,yB;MAwIA,iD;MAxIA,qC;QAKI,OAwoIO,gC AxIK,mBAAY,OAAZ,CAwIL,C;O;KA7IX,C;iFAQA,yB;MAwIA,iD;MAxIA,qC;QAKI,OAwoIO,gCAxIK,kBAAa, OAAb,EAwIL,C;O;KA7IX,C;gFAQA,yB;MAwIA,iD;MAxIA,qC;QAKI,OAwoIO,gCAxIK,kBAAC,OAAAd,EAwIL, C;O;KA7IX,C;iFAQA,yB;MAwIA,iD;MAxIA,qC;QAKI,OAwoIO,gCAxIK,sBAAE,OAaf,CAwIL,C;O;KA7IX,C;iF AQA,yB;MAwIA,iD;MAxIA,qC;QAKI,OAwoIO,gCAxIK,mBAAY,OAAZ,CAwIL,C;O;KA7IX,C;IAQA,sC;MAKI ,OAAO,oBAAoB,SAAPB,EAA0B,QAA1B,C;K;IAGX,sC;MAII,OAAO,mBAAwB,UAAL,SAAK,EAAO,mBAAO ,QAAS,KAahB,IAAP,CAAxB,EAAsD,SAAK,OAA3D,EAAiE,QAAjE,C;K;IAGX,sC;MAII,OAAO,mBAAwB,U AAL,SAAK,EAAO,mBAAO,QAAS,KAahB,IAAP,CAAxB,EAAsD,SAAK,OAA3D,EAAiE,QAAjE,C;K;IAGX,s C;MAII,OAAO,mBAAwB,UAAL,SAAK,EAAO,mBAAO,QAAS,KAahB,IAAP,CAAxB,EAAsD,SAAK,OAA3D, EAAiE,QAAjE,C;K;IAGX,sC;MAII,OAAO,oBAAoB,SAAPB,EAA0B,QAA1B,C;K;IAGX,sC;MAII,OAAO,mBA AwB,UAAL,SAAK,EAAO,mBAAO,QAAS,KAahB,IAAP,CAAxB,EAAsD,SAAK,OAA3D,EAAiE,QAAjE,C;K;I AGX,sC;MAII,OAAO,mBAAwB,UAAL,SAAK,EAAO,mBAAO,QAAS,KAahB,IAAP,CAAxB,EAAsD,SAAK,O AA3D,EAAiE,QAAjE,C;K;IAGX,sC;MAII,OAAO,oBAAoB,SAAPB,EAA0B,QAA1B,C;K;IAGX,sC;MAII,OAA O,mBAAwB,UAAL,SAAK,EAAO,mBAAO,QAAS,KAahB,IAAP,CAAxB,EAAsD,SAAK,OAA3D,EAAiE,QAAj E,C;K;iFAGX,+B;MAKI,OAAy,SAAY,QAAO,QAAP,C;K;iFAG5B,yB;MAAA,iD;MAAA,sC;QAKI,OAAO,qBA AqB,SAArB,EAA2B,QAA3B,C;O;KALX,C;iFAQA,yB;MAAA,iD;MAAA,sC;QAKI,OAAO,qBAAqB,SAArB,EA A2B,QAA3B,C;O;KALX,C;iFAQA,yB;MAAA,iD;MAAA,sC;QAKI,OAAO,qBAAqB,SAArB,EAA2B,QAA3B,C; O;KALX,C;iFAQA,yB;MAAA,iD;MAAA,sC;QAKI,OAAO,qBAAqB,SAArB,EAA2B,QAA3B,C;O;KALX,C;iFA QA,yB;MAAA,iD;MAAA,sC;QAKI,OAAO,qBAAqB,SAArB,EAA2B,QAA3B,C;O;KALX,C;iFAQA,yB;MAAA,i D;MAAA,sC;QAKI,OAAO,qBAAqB,SAArB,EAA2B,QAA3B,C;O;KALX,C;iFAQA,yB;MAAA,iD;MAAA,sC;Q AKI,OAAO,qBAAqB,SAArB,EAA2B,QAA3B,C;O;KALX,C;iFAQA,yB;MAAA,iD;MAAA,sC;QAKI,OAAO,qB AAqB,SAArB,EAA2B,QAA3B,C;O;KALX,C;8FAQA,8B;MAKI,OAAy,SAAY,QAAO,CAAQ,OAAR,CAAP,C; K;IAoBL,2B;MAAsB,OAAA,CAAE,iBAAU,CAAV,C;K;IAP/C,2B;MAOI,IAAI,mBAAO,CAAX,C;QAwQY,eAx QO,WAwQP,C;Q;IANhB,2B;MAQI,IAAI,mBAAO,CAAX,C;QAAC,UA AU,SAAV,C;K;IAGIB,wC;MAQI,IAAI, mBAAO,CAAX,C;QAAC,cAAc,SAAd,EAAoB,UAApB,C;K;IAGIB,gD;MAewD,yB;QAAA,YAAiB,C;MAAG,uB ;QAAA,UAAe,gB;MACvF,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,gBAAtC,C;MACb,gBAAC,SAAd,E AAoB,SAAPB,EAA+B,OAA/B,EAAwC,cAAxC,C;K;IAGJ,gD;MAAiC,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,U AAe,gB;MACjE,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,gBAAtC,C;MACb,eAAoB,SAAY,UAAS,SA AT,EAAoB,OAAPB,C;MACvB,KAAT,QAAS,C;K;IAGb,gD;MAakC,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,U AAe,gB;MACjE,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,gBAAtC,C;MACb,eAAoB,SAAY,UAAS,SA AT,EAAoB,OAAPB,C;MACvB,KAAT,QAAS,C;K;IAGb,gD;MAagC,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,U AAe,gB;MAC/D,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,gBAAtC,C;MACb,eAAoB,SAAY,UAAS,SA AT,EAAoB,OAAPB,C;MACvB,KAAT,QAAS,C;K;IAGb,gD;MAaiC,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,U AAe,gB;MACHe,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,gBAAtC,C;MACb,gBAAC,SAAd,EAA8C,SA A9C,EAAYD,OAAzD,EAaKE,cAAIE,C;K;IAGJ,gD;MAakC,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,gB;M ACjE,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,gBAAtC,C;MACb,eAAoB,SAAY,UAAS,SAAT,EAAoB ,OAAPB,C;MACvB,KAAT,QAAS,C;K;IAGb,gD;MAamC,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,gB;M ACIE,oCAAA,2BAAkB,SAAlB,EAA6B,OAA7B,EAAsC,gBAAtC,C;MACb,eAAoB,SAAY,UAAS,SAAT,EAAoB ,OAAPB,C;MACvB,KAAT,QAAS,C;K;IAGb,gD;MAaiC,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,gB;MA

ChE,oCAAa,2BAaKB,SAaIB,EAA6B,OAA7B,EAA5C,gBAAtC,C;MACb,eAAoB,SAAY,UAAS,SAAT,EAAoB, OAApB,C;MACvB,KAAT,QAAS,C;K;iFAGhB,iC;MAOI,SAAY,MAAK,UAAL,C;K;iFAGhB,iC;MAOI,SAAY,M AAK,UAAL,C;K;iFAGhB,iC;MAOI,SAAY,MAAK,UAAL,C;K;iFAGhB,iC;MAOI,SAAY,MAAK,UAAL,C;K;iF AGhB,iC;MAOI,SAAY,MAAK,UAAL,C;K;iFAGhB,iC;MAOI,SAAY,MAAK,UAAL,C;K;iFAGhB,iC;MAOI,SA AY,MAAK,UAAL,C;K;IAGhB,yC;MAMI,IAAI,mBAAO,CAAX,C;QAAC,gBAAC,SAAd,EAAoB,UAApB,C;K;I AGIB,+D;MAa0E,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,gB;MACzG,oCAAa,2BAaKB,SAaIB,EAA6B, OAA7B,EAA5C,gBAAtC,C;MACb,gBAAC,SAAd,EAAoB,SAApB,EAA+B,OAA/B,EAAwC,UAAxC,C;K;IAGJ, mC;MAII,OAAO,EAAS,MAAM,MAAK,SAAL,C;K;IAG1B,mC;MAII,OAAO,EAAS,MAAM,MAAK,SAAL,C;K ;IAG1B,mC;MAII,OAAO,EAAS,MAAM,MAAK,SAAL,C;K;IAG1B,mC;MAII,OAAO,EAAS,MAAM,MAAK,S AAL,C;K;IAG1B,mC;MAII,OAAO,EAAS,MAAM,MAAK,SAAL,C;K;IAG1B,mC;MAII,OAAO,EAAS,MAAM, MAAK,SAAL,C;K;IAG1B,mC;MAII,OAAO,EAAS,MAAM,MAAK,SAAL,C;K;IAOH,kD;MAAA,wB;QAAW,q CAAK,KAAL,E;O;K;IAJIC,oC;MAII,OAAO,iBAAM,gBAAN,EAAY,gCAAZ,C;K;IuDnpEX,oB;MAAA,wB;MA EI,6B;MACA,gC;MAKuB,UAAT,MAAS,EAAT,MAAS,EAAT,M;MAFV,eAAe,kE;MACf,iBAAiB,eAAS,GAAT, C;MACE,sBAAT,QAAS,C;MAAT,mB;MAAA,kB;MAAA,kB;MAAV,8C;QACI,WAAW,oBAAS,CAAT,CzC2Bu B,IyC3BIC,IAA+B,C;;MAInC,qBAAqB,48C;MACrB,WAAW,mBAAmB,cAAAnB,EAAmC,UAAAnC,EAA+C,IAA/ C,C;MACX,YAAy,eAAS,IAAK,OAAI,GAAY,CAAZ,IAAT,C;MACZ,0BAAU,IAAV,e;QACI,MAAM,MAAI,C AAJ,IAAN,IAAe,MAAM,GAAN,IAAW,KAAK,GAAL,CAAX,I;;MAEnB,yBAAoB,K;MAGpB,oBAAoB,m/D;M ACpB,4BAAuB,mBAAmB,aAAAnB,EAAkC,UAAIC,EAA8C,IAA9C,C;K;;IAvB/B,gC;MAAA,+B;QAAA,c;OAA A,wB;K;IA2BA,qC;MAKkB,IAJP,I;MACH,WAAO,EAAP,C;QAAe,W;WACf,WAAO,IAAP,C;QAAgB,OAAI,C AAC,KAAO,CAAR,MAAc,CAaIB,GAAqB,QAAS,CAA9B,GAAqC,OAAS,E;;QAEID,QAAM,KAAK,CAAX,C; eACI,C;YAAK,eAAS,E;YAAAd,K;eACA,C;YAAK,OAAC,QAAS,CAAV,GAAiB,E;YAAtB,K;kBACQ,cAAS,E;Y AHRB,K;;MAJR,W;K;IAYJ,qC;MAII,SAAS,SzCRiC,I;MyCUIC,YAAy,kBAaKB,sBAAS,kBAA3B,EAA8C,EA A9C,C;MACZ,YAAy,sBAAS,kBAAT,CAA2B,KAA3B,C;MACZ,WAAW,sBAAS,qBAAT,CAA8B,KAA9B,C;M ACX,YAAy,kBAaKB,IAaIB,EAAwB,KAAK,KAAL,IAAxB,C;MAEZ,OAAW,UAAS,EAAb,GAAYC,mDAAzC, GAAoD,K;K;IAG/D,8D;MAKiB,UAIE,M;MARf,aAAa,eAAS,YAAT,C;MACb,YAAy,C;MACZ,UAAU,C;MAC V,YAAy,C;MACC,yB;MAAb,OAAa,cAAb,C;QAAa,iC;QACT,aAAa,WAAW,IzCxBc,IyCwBzB,C;QACb,MAA M,MAAQ,CAAC,SAAW,EAAZ,KAA5B,K;QACpC,IAAI,SAAS,EAAb,C;UACI,OAAO,cAAP,EAAO,sBAAP,Y AAKB,G;UACIB,MAAM,C;UACN,QAAQ,C;;UAER,gBAAS,CAAT,I;;MAGR,OAAO,M;K;ICIEY,+B;MAII,eA Ae,CAAC,iBAAO,CAAP,IAAD,IAAa,CAAb,I;MACf,IAAI,WAAW,CAAf,C;QAAkB,M;MACIB,mBAAmB,2B; MACnB,iBAAc,CAAd,WAAiB,QAAjB,U;QACI,UAAU,sBAAK,KAAL,C;QACV,sBAAK,KAAL,EAAc,sBAAK, YAAL,CAAd,C;QACA,sBAAK,YAAL,EAAqB,GAARb,C;QACA,mC;;K;IrDbR,wB;MAOI,OAAW,oBAAK,CAA L,MAAJ,GAAY,CAAZ,GAAMb,C;K;mFAG9B,yB;MAkBA,iB;MAiBA,uB;QAMI,OAKBO,MAAO,KAIBC,CAk BD,EAIBY,CAkBZ,C;O;KAXBIB,C;mFASA,yB;MASA,iB;MATA,uB;QAMI,OASO,MAAO,KATC,CASD,EATY ,CASZ,C;O;KAFiB,C;mFASA,yB;MAAA,iB;MAAA,uB;QAMI,OAAO,MAAO,KAAI,CAAJ,EAAO,CAAP,C;O;K ANIB,C;mFASA,gB;MAMI,OAAW,kBAAK,CAAL,MAAJ,GAAY,CAAZ,GAAMb,C;K;mFAG9B,yB;MAAA,iB; MAAA,uB;QAQI,OAAO,MAAO,KAAI,CAAJ,EAAO,CAAP,C;O;KARIB,C;mFAWA,yB;MAAA,iB;MAAA,uB; QAQI,OAAO,MAAO,KAAI,CAAJ,EAAO,CAAP,C;O;KARIB,C;IAWA,2B;MAOI,OAAO,SAAM,CAAN,EAAS, SAAM,CAAN,EAAS,CAAT,CAAT,C;K;mFAGX,yB;MAAA,iB;MAAA,0B;QAMI,OAAO,MAAO,KAAI,CAA N,EAAiB,CAAJ,EAA4B,CAA5B,C;O;KANIB,C;mFASA,yB;MAAA,iB;MAAA,0B;QAMI,OAAO,MAAO,KAA M,CAAN,EAAiB,CAAJ,EAA4B,CAA5B,C;O;KANIB,C;mFASA,yB;MAAA,iB;MAAA,0B;QAMI,OAAO,MAA O,KAAI,CAAJ,EAAO,CAAP,EAAU,CAAV,C;O;KANIB,C;mFASA,mB;MAMW,UAAe,CAPeX,iBAoEc,CAPeD ,MAAJ,GAoEe,CAPeF,GAoEkB,C;MAAZB,OAAa,CAPeF,iBAAK,GAAL,MAAJ,GAoEM,CAPeN,GAAMb,G;K; mFAuE9B,yB;MAAA,iB;MAAA,0B;QAQI,OAAO,MAAO,KAAI,CAAJ,EAAO,CAAP,EAAU,CAAV,C;O;KARI B,C;mFAWA,yB;MAAA,iB;MAAA,0B;QAQI,OAAO,MAAO,KAAI,CAAJ,EAAO,CAAP,EAAU,CAAV,C;O;KA RIB,C;IAWA,4B;MAQc,Q;MADV,UAAU,C;MACV,wBAAU,KAAV,gB;QAAU,QAAA,KAAV,M;QAAiB,MAA M,SAAM,GAAN,EAAW,CAAX,C;;MACvB,OAAO,G;K;IAGX,4B;MAMc,Q;MADV,UAAU,C;MACV,wBAAU, KAAV,gB;QAAU,QAAA,KAAV,M;QAAiB,MAxHV,MAAO,KAwHe,GAXHf,EAwHoB,CAXHpB,C;;MAyHd,O AAO,G;K;IAGX,4B;MAMc,Q;MADV,UAAU,C;MACV,wBAAU,KAAV,gB;QAAU,QAAA,KAAV,M;QAAiB,M

AIIV,MAAO,KAkIe,GAlIf,EaKIoB,CAlIpB,C;;MAMId,OAAO,G;K;IAGX,4B;MAMc,Q;MADV,UAAU,C;MAC V,wBAAU,KAAV,gB;QAAU,QAAA,KAAV,M;QAAiB,MA5IV,MAAO,KA4Ie,GA5If,EA4IoB,CA5IpB,C;;MA6I d,OAAO,G;K;IAGX,4B;MAMc,Q;MADV,UAAU,C;MACV,wBAAU,KAAV,gB;QAAU,QAAA,KAAV,M;QAAu B,UAAAM,G;QAAZ,MA7IN,oBA6IuB,CA7IvB,MAAJ,GAAY,GA AZ,GA6I2B,C;;MACIC,OAAO,G;K;IAGX,4B; MAQc,Q;MADV,UAAU,C;MACV,wBAAU,KAAV,gB;QAAU,QAAA,KAAV,M;QAAiB,MA9IV,MAAO,KA8Ie, GA9If,EA8IoB,CA9IpB,C;;MA+Id,OAAO,G;K;IAGX,4B;MAQc,Q;MADV,UAAU,C;MACV,wBAAU,KAAV,gB ;QAAU,QAAA,KAAV,M;QAAiB,MA/IV,MAAO,KA+Ie,GA/If,EA+IoB,CA/IpB,C;;MAGJd,OAAO,G;K;IAGX,w B;MAOI,OAAW,oBAAK,CAAL,MAAJ,GAAY,CAAZ,GAAMb,C;K;mFAG9B,yB;MAkBA,iB;MAIbA,uB;QAM I,OakBO,MAAO,KAIBC,CAkBD,EAIBY,CAkBZ,C;O;KAXBIB,C;mFASA,yB;MASA,iB;MATA,uB;QAMI,OAS O,MAAO,KATC,CASD,EATY,CASZ,C;O;KafIb,C;mFASA,yB;MAAA,iB;MAAA,uB;QAMI,OAAO,MAAO,K AAI,CAAJ,EAAO,CAAP,C;O;KANIB,C;mFASA,gB;MAMI,OAAW,kBAAK,CAAL,MAAJ,GAAY,CAAZ,GAA mb,C;K;mFAG9B,yB;MAAA,iB;MAAA,uB;QAQI,OAAO,MAAO,KAAI,CAAJ,EAAO,CAAP,C;O;KARIB,C;mF AWA,yB;MAAA,iB;MAAA,uB;QAQI,OAAO,MAAO,KAAI,CAAJ,EAAO,CAAP,C;O;KARIB,C;IAWA,2B;MA OI,OAAO,SAAM,CAAN,EAAS,SAAM,CAAN,EAAS,CAAT,CAAT,C;K;mFAGX,yB;MAAA,iB;MAAA,0B;QA MI,OAAO,MAAO,KAAM,CAAN,EAAiB,CAAjB,EAA4B,CAA5B,C;O;KANIB,C;mFASA,yB;MAAA,iB;MAAA ,0B;QAMI,OAAO,MAAO,KAAM,CAAN,EAAiB,CAAjB,EAA4B,CAA5B,C;O;KANIB,C;mFASA,yB;MAAA,iB ;MAAA,0B;QAMI,OAAO,MAAO,KAAI,CAAJ,EAAO,CAAP,EAAU,CAAV,C;O;KANIB,C;mFASA,mB;MAM W,UAAe,CAPeX,iBAoEc,CAPeD,MAAJ,GAoEe,CAPeF,GAoEkB,C;MAAzB,OAAa,CAPeF,iBAAK,GAAL,MA AJ,GAoEM,CAPeN,GAAMb,G;K;mFAuE9B,yB;MAAA,iB;MAAA,0B;QAQI,OAAO,MAAO,KAAI,CAAJ,EAA O,CAAP,EAAU,CAAV,C;O;KARIB,C;mFAWA,yB;MAAA,iB;MAAA,0B;QAQI,OAAO,MAAO,KAAI,CAAJ,E AAO,CAAP,EAAU,CAAV,C;O;KARIB,C;IAWA,4B;MAQc,Q;MADV,UAAU,C;MACV,wBAAU,KAAV,gB;QA AU,QAAA,KAAV,M;QAAiB,MAAM,SAAM,GAAN,EAAW,CAAX,C;;MACvB,OAAO,G;K;IAGX,4B;MAMc,Q ;MADV,UAAU,C;MACV,wBAAU,KAAV,gB;QAAU,QAAA,KAAV,M;QAAiB,MAxHV,MAAO,KAwHe,GAXH f,EAwHoB,CAXHpB,C;;MAyHd,OAAO,G;K;IAGX,4B;MAMc,Q;MADV,UAAU,C;MACV,wBAAU,KAAV,gB; QAAU,QAAA,KAAV,M;QAAiB,MAIIV,MAAO,KAkIe,GAlIf,EaKIoB,CAlIpB,C;;MAMId,OAAO,G;K;IAGX,4B ;MAMc,Q;MADV,UAAU,C;MACV,wBAAU,KAAV,gB;QAAU,QAAA,KAAV,M;QAAiB,MA5IV,MAAO,KA4I e,GA5If,EA4IoB,CA5IpB,C;;MA6Id,OAAO,G;K;IAGX,4B;MAMc,Q;MADV,UAAU,C;MACV,wBAAU,KAAV ,gB;QAAU,QAAA,KAAV,M;QAAuB,UAAAM,G;QAAZ,MA7IN,oBA6IuB,CA7IvB,MAAJ,GAAY,GA AZ,GA6I2B ,C;;MACIC,OAAO,G;K;IAGX,4B;MAQc,Q;MADV,UAAU,C;MACV,wBAAU,KAAV,gB;QAAU,QAAA,KAAV, M;QAAiB,MA9IV,MAAO,KA8Ie,GA9If,EA8IoB,CA9IpB,C;;MA+Id,OAAO,G;K;IAGX,4B;MAQc,Q;MADV,U AAU,C;MACV,wBAAU,KAAV,gB;QAAU,QAAA,KAAV,M;QAAiB,MA/IV,MAAO,KA+Ie,GA/If,EA+IoB,CA/ IpB,C;;MAGJd,OAAO,G;K;IsDvaX,iB;MAAA,qB;MAEI,0BAA0B,gBACtB,EADsB,EACd,IADc,EACN,IADM,EA CE,IADF,EACU,IADV,EACkB,IADIB,EAC0B,IAD1B,EACkC,IADIC,EAC0C,IAD1C,EACkD,IADID,EAC0D,I AD1D,EACkE,IADIE,EAC0E,IAD1E,EACkF,IADIF,EAC0F,IAD1F,EACkG,IADIG,EAC0G,IAD1G,EACkH,IAD IH,EAC0H,IAD1H,EACkI,IADII,EAETB,IAFsB,EAEd,IAFc,EAEN,IAFM,EAEE,IAFF,EAEU,IAFV,EAEB,IAFI B,EAE0B,IAF1B,EAekC,IAFIC,EAE0C,IAF1C,EAekD,KAFID,EAE0D,KAF1D,EAekE,KAFIE,EAE0E,KAF1E, EAekF,KAFIF,EAE0F,KAF1F,EAekG,KAFIG,EAE0G,KAF1G,E;K;;;IAF9B,6B;MAAA,4B;QAAA,W;OAAA,qB ;K;IAQA,0C;MAKI,aAAa,C;MACb,UAAU,KAAM,OAAN,GAAa,CAAb,I;MACV,aAAa,E;MACb,YAAAY,C;MA CZ,OAAO,UAAU,GAAjB,C;QACI,SAAS,CAAC,SAAS,GAAT,IAAD,IAAiB,CAAjB,I;QACT,QAAQ,MAAM,M AAN,C;QACR,IAAI,SAAS,KAAb,C;UACI,SAAS,SAAS,CAAT,I;aACR,IAAI,WAAU,KAAAd,C;UACD,OAAO,M ;;UAEP,MAAM,SAAS,CAAT,I;;MAEd,OAAO,UAAc,SAAS,KAAb,GAAoB,CAAPB,GAA2B,CAArC,K;K;IAG X,mC;MAKI,SAAS,S3CCiC,I;M2CA1C,YAAAY,kBAaKB,mBAAM,mBAAXB,EAAoC,EAApC,C;MACZ,WA AW,KAAK,mBAAM,mBAAN,CAAiB,KAAjB,CAAL,I;MACX,OAAW,OAAO,EAAX,GAAe,IAAf,GAAyB,E;K;IAG pC,gC;MAIL,OAAO,6BAAoB,C;K;IC7C/B,kB;MAAA,sB;MAEI,6B;MACA,8B;MACA,gC;MAKuB,UAAT,MAA S,EAAT,MAAS,EAAT,M;MAFV,eAAe,kE;MACf,iBAAiB,eAAS,GAAT,C;MACE,sBAAT,QAAS,C;MAAT,mB; MAAA,kB;MAAA,kB;MAAV,8C;QACI,WA AW,oBAAS,CAAT,C5C0BuB,I4C1BIC,IAA+B,C;;MAInC,qBAAq B,sW;MACrB,WA AW,mBAAMb,cAAnB,EAAMc,UAAnc,EAA+C,GAA/C,C;MACX,YAAAY,eAAS,IAAK,OAA d,C;MACZ,0BAAU,IAAV,e;QACI,IAAI,QA AK,CAAT,C;UAAY,MAAM,GAAN,IAAW,KAAK,GAAL,C;;UACI

B,MAAM,GAAN,IAAW,MAAM,MAAI,CAAJ,IAAN,IAAe,KAAK,GAAL,CAAf,I;;MAEpB,yBAAoB,K;MAGpB,kBAAkB,0U;MACIB,0BAAqB,mBAAmB,WAAAnB,EAAGC,UAAhC,EAA4C,GAA5C,C;MAGrB,oBAAoB,i8B;MACpB,4BAAuB,mBAAmB,aAAnB,EAakC,UAAIC,EAA8C,GAA9C,C;K;;;IA7B/B,8B;MAAA,6B;QAAA,Y;OAAA,sB;K;IAiCA,iC;MAII,OAAO,6BAAmB,C;K;IAG9B,oC;MAIW,wCAAmB,C;MAAnB,U;QAA6B,wB5CRM,a4CQN,C;OAApC,W;K;IAGJ,oC;MAIW,wCAAmB,C;MAAnB,U;QAA6B,wB5CfM,a4CeN,C;OAApC,W;K;IAGJ,kC;MAQI,SAAS,S5C1BiC,I;M4C2B1C,YAAy,kBAAkB,oBAAO,kBAAzB,EAA4C,EAA5C,C;MAEZ,iBAAiB,oBAAO,kBAAP,CAAyB,KAAzB,C;MACjB,eAAe,aAAa,oBAAO,mBAAP,CAA0B,KAA1B,CAAb,GAAgD,CAAhd,I;MACf,WAAW,oBAAO,qBAAP,CAA4B,KAA5B,C;MAEX,IAAI,KAAK,QAAT,C;QACI,OAAO,C;OAGX,kBAAkB,OAAS,C;MAE3B,IAAI,gBAAe,CAAnB,C;QACI,YAAy,C;QACZ,gBAAgB,U;QACHB,AAAU,CAAV,OAAa,CAAb,M;UACI,yBAAc,QAAS,KAAV,GAAqB,GAAIC,K;UACA,IAAI,YAAy,EAAhB,C;YACI,OAAO,C;WAEX,gBAAS,CAAT,I;UACA,yBAAc,QAAS,KAAV,GAAqB,GAAIC,K;UACA,IAAI,YAAy,EAAhB,C;YACI,OAAO,C;WAEX,gBAAS,CAAT,I;;QAEJ,OAAO,C;OAGX,IAAI,QAAQ,CAAZ,C;QACI,OAAO,W;OAGX,eAAgB,KAAK,UAAL,I;MACHB,cAAgB,QAAQ,EAaz,GAAkB,WAAW,CAA7B,GAAoC,Q;MACHD,OAAQ,SAAU,IAAI,OAaj,IAAV,CAAD,GAA2B,C;K;ICnGtC,0B;MAAA,8B;MACI,+BAA+B,gBAC3B,GAD2B,EACnB,GADmB,EACX,GADW,EACH,GADG,EACK,GADL,EACa,GADb,EACqB,GADrB,EAC6B,IAD7B,EACqC,IADrC,EAC6C,IAD7C,EACqD,IADrD,EAC6D,IAD7D,EACqE,IADrE,EAC6E,IAD7E,EACqF,IADrF,EAC6F,KAD7F,EACqG,KADrG,EAC6G,KAD7G,EACqH,KADrH,EAC6H,KAD7H,E;MAG/B,gCAAgC,gBAC5B,CAD4B,EACzB,CADyB,EACtB,CADsB,EACnB,CADmB,EACHB,CADgB,EACb,CADa,EACV,CADU,EACP,EADO,EACH,CADG,EACA,EADA,EACI,CADJ,EACO,CADP,EACU,EADV,EACc,EADd,EACKb,EADiB,EACsB,CADtB,EACyB,CADzB,EAC4B,CAD5B,EAC+B,CAD/B,EACkC,CADIC,E;K;;;IAJpC,sC;MAAA,qC;QAAA,oB;OAAA,8B;K;IASA,qC;MACI,YAAy,kBAAkB,4BAAe,wBAajC,EAakD,SAaID,C;MACZ,OAAO,SAAS,CAAT,IAAc,aAAO,4BAAe,wBAAf,CAA+B,KAA/B,IAAwC,4BAAe,yBAaf,CAAgC,KAAhC,CAAxC,IAAP,C;K;ICXzB,qC;MACI,OA Ae,IAAR,8BAAgB,IAAhB,KACY,IAAR,8BAAgB,IADpB,C;K;ICCX,wC;M5CiBW,Q;MAAA,I4CXgB,K5CWZ,IAAS,CAAT,I4CXY,K5CWE,IAAS,2BAA3B,C;QAAA,OAAsC,qB4CXtB,K5CWsB,C;;Q4CXb,MAAM,8BAA0B,mCAAyB,gBAAzB,MAA1B,C;;MAAtC,W;K;ICRJ,sC;MAEI,WAAW,ShDkC+B,I;MgDhC1C,IAAY,GAAR,oBAAgB,GAAhB,KAAkC,GAAR,oBAAgB,GAA1C,CAAJ,C;QACI,OAA8B,OAAtB,KAAK,CAAC,OAAO,CAAP,IAAD,IAAa,CAAb,IAAL,KAAsB,C;OAGIC,IAAY,IAAR,oBAAgB,IAAhB,KAAkC,IAAR,oBAAgB,IAA1C,CAAJ,C;QACI,OAAO,S;OAEX,OAAO,wB;K;ICPX,wC;MxCqTe,WwC7SY,KxC6SZ,IAAS,C;MAAT,S;QAAC,OwC7SF,KxC6SE,IAqgHT,gBAAR,iBAAQ,C;OArgHT,U;MAAA,S;QAAA,SAAsC,sBwC7StB,KxC6SsB,C;;QwC7Sb,MAAM,8BAA0B,iCAAuB,cAAvB,MAA1B,C;;MAAtC,a;K;IAGJ,wC;MxCsTe,WwC9SY,KxC8SZ,IAAS,C;MAAT,S;QAAC,OwC9SF,KxC8SE,IAigHT,gBAAR,iBAAQ,C;OajgHT,U;MAAA,S;QAAA,SAAsC,sBwC9StB,KxC8SsB,C;;QwC9Sb,MAAM,8BAA0B,iCAAuB,cAAvB,MAA1B,C;;MAAtC,a;K;IAGJ,wC;MxCuTe,WwC/SY,KxC+SZ,IAAS,C;MAAT,S;QAAC,OwC/SF,KxC+SE,IA6/GT,gBAAR,iBAAQ,C;OA7/GT,U;MAAA,S;QAAA,SAAsC,sBwC/StB,KxC+SsB,C;;QwC/Sb,MAAM,8BAA0B,iCAAuB,cAAvB,MAA1B,C;;MAAtC,a;K;IAGJ,wC;MxCwTe,WwChTY,KxCgTZ,IAAS,C;MAAT,S;QAAC,OwChTF,KxCgTE,IAy/GT,gBAAR,iBAAQ,C;OAz/GT,U;MAAA,S;QAAA,SAAsC,sBwChTtB,KxCgTsB,C;;QwChTb,MAAM,8BAA0B,iCAAuB,cAAvB,MAA1B,C;;MAAtC,a;K;IASO,6C;MAAA,8B;MAAS,uB;K;8FACW,Y;MAAQ,OAAA,gBAAY,K;K;+CAC3C,Y;MAAkC,OAAA,gBAAY,U;K;sDAC9C,mB;MAAGD,OAAA,gBAAY,gBAAS,OAAT,C;K;mDAC5D,iB;MACI,oCAAA,2BAAkB,KAAIB,EAAYB,SAAZB,C;MACb,OAAO,6BAAY,KAAZ,C;K;qDAEX,mB;MAES,Q;MAAL,IAAI,eAAC,0EAAD,OAAJ,C;QAAGC,OAAO,E;MACvC,OxCsrBO,UwCtrBA,gBxCsrBR,QAAQ,EwCtrBoB,O3EgOF,KmCsdIB,C;K;yDwCprBX,mB;MAES,Q;MAAL,IAAI,eAAC,0EAAD,OAAJ,C;QAAGC,OAAO,E;MACvC,OxCy6BO,cwCz6BA,gBxCy6BR,QA AQ,EwCz6BwB,O3E2NN,KmC8sBIB,C;K;;IwC/7BnB,6B;MAMI,4C;K;IA2BO,6C;MAAA,8B;MAAS,uB;K;8FACW,Y;MAAQ,OAAA,gBAAY,K;K;+CAC3C,Y;MAAkC,OAAA,gBAAY,U;K;sDAC9C,mB;MAAiD,OAAA,gBAAY,gBAAS,OAAT,C;K;mDAC7D,iB;MACI,oCAAA,2BAAkB,KAAIB,EAAYB,SAAZB,C;MACb,OAAO,6BAAY,KAAZ,C;K;qDAEX,mB;MAES,Q;MAAL,IAAI,eAAC,0EAAD,OAAJ,C;QAAiC,OAAO,E;MACxC,OxCqqBO,UwCrqBA,gBxCqqBR,QAAQ,EwCrqBoB,O3DgNA,KmBqdpB,C;K;yDwCnqBX,mB;MAES,Q;MAAL,IAAI,eAAC,0EAAD,OAAJ,C;QAAiC,OAAO,E;MACxC,OxCw5BO,cwCx5BA,gBxCw5BR,QAAQ,EwCx5BwB,O3D2MJ,KmB6sBpB,C;K;;IwC96BnB,6B;MAMI,4C;K;IA2BO,6C;MAAA,8B;MAAS,uB;K;8FACW,Y;MAAQ,OAAA,gBA

AY,K;K;+CAC3C,Y;MAAkC,OAAA,gBAAY,U;K;sDAC9C,mB;MAAiD,OAAA,gBAAY,gBAAS,OAAT,C;K;m  
DAC7D,iB;MACI,oCAAA,2BAAkB,KAAIB,EAAYB,SAAZB,C;MACb,OAAO,6BAAY,KAAZ,C;K;qDAEX,mB;  
MAES,Q;MAAL,IAAI,eAAC,0EAAD,QAAJ,C;QAAiC,OAAO,E;MACxC,OxCopBO,UwCppBA,gBxCopBR,QA  
AQ,EwCppBoB,O5EkIA,KoCkhBpB,C;K;yDwClpBX,mB;MAES,Q;MAAL,IAAI,eAAC,0EAAD,QAAJ,C;QAAi  
C,OAAO,E;MACxC,OxCu4BO,cwCv4BA,gBxCu4BR,QAAQ,EwCv4BwB,O5E6HJ,KoC0wBpB,C;K;;IwC75BnB  
,8B;MAMI,4C;K;IA2BO,6C;MAAA,8B;MAAS,uB;K;8FACW,Y;MAAQ,OAAA,gBAAY,K;K;+CAC3C,Y;MAAk  
C,OAAA,gBAAY,U;K;sDAC9C,mB;MAAkD,OAAA,gBAAY,gBAAS,OAAT,C;K;mDAC9D,iB;MACI,oCAAA,2  
BAAkB,KAAIB,EAAYB,SAAZB,C;MACb,OAAO,6BAAY,KAAZ,C;K;qDAEX,mB;MAES,Q;MAAL,IAAI,eAAC  
,0EAAD,SAAJ,C;QAAkC,OAAO,E;MACzC,OxCmoBO,UwCnoBA,gBxCmoBR,QAAQ,EwCnoBoB,O1EkHE,Kk  
CihBtB,C;K;yDwCjoBX,mB;MAES,Q;MAAL,IAAI,eAAC,0EAAD,SAAJ,C;QAAkC,OAAO,E;MACzC,OxCs3B  
O,cwCt3BA,gBxCs3BR,QAAQ,EwCt3BwB,O1E6GF,KkCywBtB,C;K;;IwC54BnB,8B;MAMI,4C;K;ICtIJ,qC;MAI  
I,SAAS,SID+BiC,I;MkD9B1C,OAAa,CAAN,gBAAc,EAAd,KACU,EAAN,gBAAc,EADIB,KAEL,OAAM,GAFV,  
KAGI,KAAC,IAAL,KACC,OAAM,IAAN,KACS,IAAN,gBAAc,IADjB,KAEG,OAAM,IAFT,IAGG,OAAM,IAHT  
,IAIG,OAAM,IAJT,IAKG,OAAM,IALT,IAMG,OAAM,KAPV,CAHJ,C;K;;;mCCTP,gB;;K;;ICAJ,wB;K;;IAIA,w  
B;K;;IAIA,wB;K;;IAKiC,uB;MAAC,oB;QAAA,OAA0B,E;MAA1B,gB;K;;IAEIC,kB;K;;IAqCqC,sB;MAAC,gB;K  
;;IAGCN,4B;MAAC,sB;K;;IAEjC,uB;K;;IA8DmC,4B;MAAC,kB;K;;IAEpC,oB;K;;ICpJA,oB;K;;IAIA,wB;K;;oF7  
DLA,qB;MAKqE,uCoCHtB,E;K;iGpCK/C,yB;MAAA,kD;MAAA,4B;QAQsE,mBAAY,SAAZ,C;O;KARtE,C;IAU  
A,iC;MAGI,OAAsB,UAAy,QAAvB,KAAmC,SAA9C,GACe,UAAy,UAD3B,GAGI,gBAAGB,UAAhB,C;K;IAG  
R,qC;MAEI,YoC1B2C,E;MpC2B3C,eAAe,UAAW,W;MAC1B,OAAO,QAAS,UAAhB,C;QACU,KAAY,MAAK,  
QAAS,OAAd,C;MACtB,OAAO,K;K;IAGX,8C;MAQc,Q;MANV,IAAI,KAAM,OAAN,GAAa,UAAW,KAA5B,C;  
QACI,OAAO,gBAAGB,UAAhB,C;OAEX,eAAe,UAAW,W;MAC1B,YAAy,C;MACZ,OAAO,QAAS,UAAhB,C;  
QACI,MAAM,YAAN,EAAM,oBAAN,UAAiB,QAAS,O;;MAE9B,IAAI,QAAQ,KAAM,OAAlB,C;QACI,MAAM,  
KAAN,IAAe,I;OAEhB,OAAO,K;K;IAIX,yB;MAG6C,sBAAY,OAAZ,E;K;wGAE7C,yB;MAAA,+D;MAAA,gC;Q  
AI0B,gBAAf,gB;QAAqB,aJW5B,W;QIXA,OJYO,SIZoC,Q;O;KAJ/C,C;yGAOA,yB;MAAA,4E;MAAA,gE;MAA  
A,0C;QAI,qBAAqB,QAArB,C;QAC8B,gBAAvB,eAAa,QAAb,C;QAA6B,aJGpC,W;QIHA,OJIO,SIJ4C,Q;O;KA  
LvD,C;IASA,wB;MAG2C,oBAAU,OAAV,E;K;sGAE3C,yB;MAAA,uE;MAAA,gC;QAI8B,gBAAnB,oB;QAAyB,  
aJVhC,W;QIUA,OJTO,SISwC,Q;O;KAJnD,C;wGAOA,yB;MAAA,wE;MAAA,0C;QAIsC,gBAA3B,mBAAiB,QA  
AjB,C;QAAiC,aJbXc,W;QIiBA,OJhBO,SIgBgD,Q;O;KAJ3D,C;IAQA,qB;MAIuD,oBAAU,IAAV,E;K;sGAEvD,  
yB;MAAA,wE;MAAA,gC;QAIiC,gBAAtB,oB;QAA4B,aJ/BnC,W;QI+BA,OJ9BO,SI8B2C,Q;O;KAJtD,C;uGAOA  
,yB;MAAA,uE;MAAA,0C;QAIyC,gBAA9B,mBAAoB,QAApB,C;QAAoC,aJtC3C,W;QIsCA,OJrCO,SIqCmD,Q;  
O;KAJ9D,C;IAQA,mC;MAOqB,Q;MAAA,kC;MAAjB,iBAAc,CAAd,yB;QACI,sBAAK,KAAL,EAAC,KAAd,C;;  
K;IAIR,+B;MAMuD,sBAAQ,4BAAR,C;K;IAEvD,6B;MAIwE,kBAAhB,0B;MAAwB,uB;MAAxB,OJJE7C,W;K;II  
mEX,4B;MAQI,gBAAGB,SAAhB,EAAsB,cAAtB,C;K;IAGJ,2C;MAQI,gBAAGB,SAAhB,EAAsB,UAAtB,C;K;IA  
GJ,2C;MACI,IAAI,IAAK,KAAL,IAAa,CAAJB,C;QAAoB,M;MAEpB,YAAy,YAAy,IAAZ,C;MACZ,gBAAC,KA  
Ad,EAAqB,UAArB,C;MAEA,aAAU,CAAV,MAAkB,KAAM,OAAXB,M;QACI,iBAAK,CAAL,EAAU,MAAM,C  
AAN,CAAV,C;;K;IAIR,uC;MACI,OAAO,gBAAkB,IAAlB,O;K;IAGX,iF;MAII,oCAAA,2BAAkB,UAAIB,EAA8B  
,QAA9B,EAAwC,MAAO,OAA/C,C;MACb,gBAAGB,WAAW,UAAx,I;MACHb,oCAAA,2BAAkB,iBAAIB,EAAq  
C,oBAAoB,SAApB,IAArC,EAAoE,WAAy,OAAhF,C;MAEb,IAAI,WAAkB,QAAO,WAAp,CAAIB,IAAyC,WA  
AkB,QAAO,MAAP,CAA/D,C;QACI,eAAsB,MAAY,UAAS,UAAT,EAAqB,QAArB,C;QACtB,WAAy,KAAI,QA  
AJ,EAAC,iBAAd,C;;QAExB,IAAI,WAAW,WAAx,IAA0B,qBAAqB,UAAAnD,C;UACI,iBAAC,CAAd,UAAAsB,SA  
AtB,U;YACI,YAAy,oBAAoB,KAApB,IAAZ,IAAyC,OAAO,aAAa,KAAb,IAAP,C;;;UAG7C,mBAAC,YAAy,CA  
AZ,IAAd,aAAmC,CAAnC,Y;YACI,YAAy,oBAAoB,OAAPB,IAAZ,IAAyC,OAAO,aAAa,OAAb,IAAP,C;;;K;8G  
AMzD,qB;MAEGf,gB;K;kGAehF,yB;MAAA,4D;MAAA,4B;QAC8E,OAAK,aAAL,SAAK,C;O;KADnF,C;sGAI  
A,gC;MAEI,OAAI,SAAJ,GAEL,SAFJ,GAIL,SN83BoB,Q;K;IM13B5B,mC;MAEL,IAAI,QAAQ,CAAZ,C;QACI,oB  
;OAEJ,OAAO,K;K;IAGX,mC;MAEL,IAAI,QAAQ,CAAZ,C;QACI,oB;OAEJ,OAAO,K;K;IAIX,mC;MAIqD,mB;K  
;IAErD,wC;MPzNI,IAAI,EOgOI,YAAy,CPhOhB,CAAJ,C;QACI,cO+NqB,gC;QP9NrB,MAAM,gCAAYB,OAAQ,  
WAAjC,C;Q;IOiOd,8C;MAAoE,Y;K;I8D1PV,qC;MAAiC,6B;K;uDAIvF,mB;MACI,qB;MACA,eAAe,e;MACf,O  
AAO,QAAS,UAAhB,C;QACI,IAAI,OAAA,QAAS,OAAT,EAAMB,OAAnB,CAAJ,C;UACI,QAAS,S;UACT,OAA

O,I;;MAGf,OAAO,K;K;yDAGX,oB;MAGoB,Q;MAFhB,qB;MACA,eAAe,K;MACC,0B;MAAhB,OAAgB,cAAhB ,C;QAAgB,yB;QACZ,IAAI,eAAI,OAAJ,CAAJ,C;UAAkB,WAAW,I;;MAEjC,OAAO,Q;K;IAKuC,sE;MAAA,qB; QAAE,OAAM,gBAAN,mB;O;K;4DAFpD,oB;MAEY,Q;MADR,qB;MACA,OAAoC,YAA5B,iEAA4B,EAAU,oD AAV,C;K;IAKU,sE;MAAA,qB;QAAE,QAAO,gBAAP,mB;O;K;4DAFpD,oB;MAEY,Q;MADR,qB;MACA,OAAo C,YAA5B,iEAA4B,EAAU,oDAAV,C;K;gDAGxC,Y;MACI,qB;MACA,eAAe,IAAK,W;MACpB,OAAO,QAAS,U AAhB,C;QACI,QAAS,O;QACT,QAAS,S;;K;iDAIjB,Y;MAE8B,OAAA,IAAK,U;K;yDAGnC,Y;K;;IC3CgD,+B;M AAiC,oC;MACjF,gBAA8B,C;K;8CAM9B,mB;MAMI,qB;MACA,iBAAI,SAAJ,EAAU,OAAV,C;MACA,OAAO,I ;K;mDAGX,2B;MAMc,UACF,M;MANR,oCAAA,4BAAmB,KAAAnB,EAA0B,SAAI1B,C;MAEb,qB;MACA,aAAa, K;MACb,cAAc,K;MACJ,0B;MAAV,OAAU,cAAV,C;QAAU,mB;QACN,kBAAI,eAAJ,EAAI,uBAAJ,WAAc,CA Ad,C;QACA,UAAU,I;;MAEd,OAAO,O;K;0CAGX,Y;MACI,qB;MACA,yBAAY,CAAZ,EAAe,SAAf,C;K;IAKiB, gE;MAAA,qB;QAAE,OAAM,gBAAN,mB;O;K;sDAFvB,oB;MACI,qB;MACA,OAAO,kBAAU,8CAAV,C;K;IAK U,gE;MAAA,qB;QAAE,QAAO,gBAAP,mB;O;K;sDAFvB,oB;MACI,qB;MACA,OAAO,kBAAU,8CAAV,C;K;6C AIX,Y;MAAqD,iD;K;mDAErD,mB;MAAoD,0BAAQ,OAAAR,KAAoB,C;K;kDAExE,mB;MACqB,Q;MAAA,6B; MAAjB,iBAAc,CAAd,yB;QACI,IAAI,wBAAI,KAAJ,GAAC,OAAd,CAAJ,C;UACI,OAAO,K;;MAGf,OAAO,E;K; sDAGX,mB;MACI,iBAAc,sBAAd,WAA+B,CAA/B,U;QACI,IAAI,wBAAI,KAAJ,GAAC,OAAd,CAAJ,C;UACI,O AAO,K;;MAGf,OAAO,E;K;iDAGX,Y;MAA6D,iCAAA,CAAb,C;K;yDAC7D,iB;MAAuE,sDAAiB,KAAjB,C;K;o DAGvE,8B;MAA4E,uCAAQ,IAAR,EAAc,SAAd,EAAyB,OAAzB,C;K;wDAE5E,8B;MAII,eAAe,0BAAa,SAAb, C;MACf,YAAO,UAAU,SAAV,I;MnEuDX,iBAAc,CAAd,UAAsB,KAAtB,U;QmEtDiB,e;QACA,iB;;K;2CAIjB,iB ;MAMI,IAAI,UAAU,IAAd,C;QAAoB,OAAO,I;MAC3B,IAAI,2BAAJ,C;QAAuB,OAAO,K;MAE9B,OAAO,oCA Aa,uBAAc,IAAd,EAAoB,KAApB,C;K;6CAGxB,Y;MAG+B,OAAA,oCAAA,yBAAgB,IAAhB,C;K;IAG5C,kD;M AAA,oB;MACI,eACsB,C;MACtB,cAIqB,E;K;yDAErB,Y;MAAkC,sBAAQ,gB;K;sDAE1C,Y;MAEW,Q;MADP,I AAI,CAAC,cAAL,C;QAAgB,MAAM,6B;MACtB,eAAO,mBAAP,EAAO,2BAAP,O;MACA,OAAO,wBAAI,WA AJ,C;K;wDAGX,Y;MtE5CJ,IAAI,EsE6CU,gBAAQ,EtE7CIB,CAAJ,C;QACI,csE4CwB,sE;QtE3CxB,MAAM,6BA AsB,OAAQ,WAA9B,C;OsE6CF,6BAAS,WAAAT,C;MACA,eAAQ,W;MACR,cAAO,E;K;;IAOqB,6D;MAHpC,oB; MAGmD,wD;MAG3C,oCAAA,4BAAmB,KAAAnB,EAA0B,WAAyB,KAAAnD,C;MACb,eAAa,K;K;iEAGjB,Y;MA AsC,sBAAQ,C;K;+DAE9C,Y;MAAgC,mB;K;8DAEHc,Y;MACI,IAAI,CAAC,kBAAL,C;QAAoB,MAAM,6B;MA E1B,eAAO,mCAAP,EAAO,YAAP,C;MACA,OAAO,wBAAI,WAAJ,C;K;mEAGX,Y;MAAoC,sBAAQ,CAAR,I;K ;+DAEpC,mB;MACI,wBAAI,YAAJ,EAAW,OAAX,C;MACA,mC;MACA,cAAO,E;K;+DAGX,mB;MtEIFJ,IAAI, EsEmFU,gBAAQ,EtEnFIB,CAAJ,C;QACI,csEkFwB,4E;QtEjFxB,MAAM,6BAAsB,OAAQ,WAA9B,C;OsEkFF,w BAAI,WAAJ,EAAU,OAAV,C;K;;IAIgb,+D;MAAuF,8B;MAAtF,kB;MAA0C,4B;MAC/D,eAAyB,C;MAGrB,oC AAa,2BAAkB,gBAAIB,EAA6B,OAA7B,EAAc,WAAK,KAA3C,C;MACb,eAAa,UAAU,gBAAV,I;K;wDAGjB, 0B;MACI,oCAAA,4BAAmB,KAAAnB,EAA0B,YAA1B,C;MAEb,WAAK,aAAI,mBAAY,KAAZ,IAAJ,EAAuB,OA AvB,C;MACL,mC;K;wDAGJ,iB;MACI,oCAAA,2BAAkB,KAAIB,EAAyB,YAAzB,C;MAEb,OAAO,wBAAK,mB AAY,KAAZ,IAAL,C;K;6DAGX,iB;MACI,oCAAA,2BAAkB,KAAIB,EAAyB,YAAzB,C;MAEb,aAAa,WAAK,kB AAS,mBAAY,KAAZ,IAAT,C;MACIB,mC;MACA,OAAO,M;K;wDAGX,0B;MACI,oCAAA,2BAAkB,KAAIB,EA AyB,YAAzB,C;MAEb,OAAO,WAAK,aAAI,mBAAY,KAAZ,IAAJ,EAAuB,OAAvB,C;K;mGAGO,Y;MAAQ,mB ;K;2DAE/B,Y;MAA+C,WAAK,iB;K;;;ICxMN,8B;MAAiC,sB;MAwCnF,uBAAoC,I;MA+CpC,yBAA6C,I;K;IAIF R,oD;MAAC,wB;MAGlC,gBAAqB,K;K;IFAHa,Y;MAAA,yB;K;uGAKZ,Y;MAAQ,oB;K;8DAE9B,oB;MAKI,eA Ae,IAAK,S;MACpB,gBAAc,Q;MACd,OAAO,Q;K;wDAGX,Y;MAA+B,iEAAc,IAAd,C;K;wDAC/B,Y;MAAkC,i EAAc,IAAd,C;K;sDAClC,iB;MAA4C,+DAAY,IAAZ,EAakB,KAAIB,C;K;;IAIB5C,8E;MAAA,wE;MAAsC,2CA AK,KAAM,IAAX,EAAgB,KAAM,MAAtB,C;MAAtC,Y;K;IASBJ,+C;MACsE,6B;K;mEACIE,mB;MAAmD,kCA Ac,OAAd,C;K;iEAEnD,mB;MAAiD,gCAAY,OAAZ,C;K;;yCAIrD,Y;MACI,YAAQ,Q;K;IAOQ,+F;MAAA,sD;M AAS,6B;K;uFACb,mB;MAAwC,MAAM,qCAA8B,8BAA9B,C;K;mFAC9C,Y;MACI,4BAAwB,Q;K;4FAG5B,mB ;MAAsD,sDAAY,OAAZ,C;K;IAI3C,oH;MAAA,kD;K;4GACH,Y;MAAkC,OAAA,0BAAc,U;K;yGACHd,Y;MAA yB,OAAA,0BAAc,OAAO,I;K;2GAC9C,Y;MAAwB,0BAAc,S;K;;sFAL9C,Y;MACI,oBAAoB,oCAAQ,W;MAC5 B,6G;K;0FAOJ,mB;MACI,qB;MACA,IAAI,+CAAY,OAAZ,CAAJ,C;QACI,4BAAwB,cAAO,OAAP,C;QACxB,O AAO,I;OAEX,OAAO,K;K;oIAGY,Y;MAAQ,OAAA,4BAAwB,K;K;4FAEvD,Y;MAAsC,4BAAwB,iB;K;;0FA9B 1E,Y;MACI,IAAI,4BAAJ,C;QACI,6F;OA+BJ,OAAO,mC;K;kDAKf,gB;MAEyB,Q;MADrB,qB;MACqB,OAAA,I

9E8Q2D,QAAQ,W;M8E9QxF,OAAqB,cAArB,C;QAAqB,wB;QAAf,U9EiMsD,U;Q8EjMjD,Y9E8MiD,Y;Q8E7M  
xD,iBAAI,GA AJ,EAAS,KAAT,C;;K;IAQc,iG;MAAA,sD;MAAS,oC;K;yFACf,mB;MAAwC,MAAM,qCAA8B,gC  
AA9B,C;K;qFAC9C,Y;MAAuB,4BAAwB,Q;K;8FAE/C,mB;MAAsD,wDAAc,OAA d,C;K;IAI3C,sH;MAAA,kD;  
K;8GACH,Y;MAAkC,OAAA,0BAAc,U;K;2GAC hD,Y;MAAyB,OAAA,0BAAc,OAAO,M;K;6GAC9C,Y;MAAw  
B,0BAAc,S;K;;wFAL9C,Y;MACI,oBAAoB,oCAAQ,W;MAC5B,+G;K;sIAOmB,Y;MAAQ,OAAA,4BAAwB,K;K  
;8FAEvD,Y;MAAsC,4BAAwB,iB;K;;4FAnB1E,Y;MACI,IAAI,8BAAJ,C;QACI,iG;OAOBJ,OAAO,qC;K;gDAGf,e  
;MACI,qB;MACA,WAAW,YAAQ,W;MACnB,OAAO,IAAK,UAAZ,C;QACI,YAAY,IAAK,O;QACjB,QAAQ,KA  
AM,I;QACd,IAAI,YAAO,CAAP,CAAJ,C;UACI,YAAY,KAAM,M;UACIB,IAAK,S;UACL,OAAO,K;;MAGf,OA  
AO,I;K;kDAIX,Y;K;;IC3I+C,8B;MAAiC,oC;K;0CAEhF,iB;MAMI,IAAI,UAAU,IAAd,C;QAAoB,OAAO,I;MAC3  
B,IAAI,0BAAJ,C;QAAsB,OAAO,K;MAC7B,OAAO,mCAAY,mBAAU,IAAV,EAAGB,KAAhB,C;K;4CAGvB,Y;  
MAG+B,OAAA,mCAAY,2BAAkB,IAAIB,C;K;;ICbT,0B;MAAuD,8B;MAAIC,4B;MACvD,4BAAkC,K;K;gCAk  
BIC,Y;MAEI,qB;MACA,4BAAa,I;MACb,OAAO,I;K;qCAGX,Y;K;iDAGA,uB;K;iFAG8B,Y;MAAQ,OAAA,oBA  
AM,O;K;sCAC5C,iB;MACyC,Q;MAAA,oCAAM,0BAAW,KAAX,CAAN,4D;K;sCACzC,0B;MAIW,IAAa,I;MA  
HpB,qB;MACA,0BAAW,KAAX,C;MAEoB,gBAAb,qBAAM,KAAN,C;MAAqB,qC;MAA5B,OAAO,CAAa,OtE8  
BjB,SsE9BI,2D;K;oCAGX,mB;MACI,qB;MACM,oBAA Y,MAAK,OAAL,C;MACIB,qC;MACA,OAAO,I;K;sCA  
GX,0B;MACI,qB;MACM,oBAA Y,QAAO,mCAAoB,KAApB,CAAP,EAAMC,CAAnC,EAAsC,OAAtC,C;MACIB  
,qC;K;yCAGJ,oB;MACI,qB;MACA,IAAI,QAAS,UAAb,C;QAAwB,OAAO,K;MAE/B,uBAAA,oBxEioDoB,QMjr  
D0C,YkEgDrD,QIEhDqD,CNirD1C,C;MwEhoDpB,qC;MACA,OAAO,I;K;yCAGX,2B;MACI,qB;MACA,mCAAo  
B,KAApB,C;MAEA,IAAI,UAAS,SAAb,C;QAAMB,OAAO,oBAAO,QAAP,C;MAC1B,IAAI,QAAS,UAAb,C;QA  
AwB,OAAO,K;MAE3B,IADE,KACF,e;QAAQ,OAAO,oBAAO,QAAP,C;WACf,IAFE,KA EF,O;QAAK,uBIE7Dq  
D,YkE6D7C,QIE7D6C,CNirD1C,QwEpnD6B,oBxEonD7B,C;;QwEnnDR,uBAAoC,cAA5B,oBAA4B,EA AV,CA  
AU,EAAP,CAA O,CAAY,QIE9DE,YkE8DK,QIE9DL,CkE8DF,EAA4C,cAAN,oBAAM,EAAY,KA AZ,EAAMB,S  
AA nB,CAA5C,C;;MAG5D,qC;MACA,OAAO,I;K;2CAGX,iB;MACI,qB;MACA,0BAAW,KAAX,C;MACA,qC;M  
ACA,OAAW,UAAS,sBAAb,GACG,oBAA Y,MADf,GAGG,oBAA Y,QAAO,KAAP,EAAC,CAAd,CAAIB,CAAm  
C,CAAnC,C;K;uCAGR,mB;MAEkB,Q;MADd,qB;MACc,2B;MAAd,mD;QACI,IAAI,4BAAM,KAAN,GAAgB,O  
AAhB,CAAJ,C;UACU,oBAA Y,QAAO,KAAP,EAAC,CAAd,C;UACIB,qC;UACA,OAAO,I;;MAGf,OAAO,K;K;8  
CAGX,8B;MACI,qB;MACA,qC;MACM,oBAA Y,QAAO,SAAP,EAAkB,UAAU,SAAV,IAAIB,C;K;gCAGtB,Y;M  
ACI,qB;MACA,uB9BhHuC,E;M8BiHvC,qC;K;wCAIJ,mB;MAA+C,OAAM,QAAN,oBAAM,EA AQ,OAAR,C;K;  
4CAErD,mB;MAAmD,OAAM,YAAN,oBAAM,EAAY,OAAZ,C;K;mCAEzD,Y;MAA0B,uBAAc,oBAA d,C;K;OC  
AE1B,iB;MAGe,UAGL,MAHK,EAMO,M;MAPIB,IAAI,KAAM,OAAN,GAAa,SAAjB,C;QACI,OAAO,2D;OAG  
c,gBAAXB,eAAK,SAAL,IAAK,gBAAL,yB;MxEuwBL,UAAU,SAAV,EwEvwBsC,KxEuwBtC,EAD+F,CAC/F,E  
ADoH,CACpH,EADuI,gBACvI,C;MwErwBI,IAAI,KAAM,OAAN,GAAa,SAAjB,C;QACI,MAAM,SAAN,IAAc,6  
E;OAGIB,OAAO,K;K;kCAGX,Y;MACI,OAAO,EAAS,MAAM,MAAK,oBAAL,C;K;yCAI1B,Y;MACI,IAAI,yBA  
AJ,C;QAAGB,MAAM,oC;K;+CAG1B,iB;MACI,oCAAa,kCAAyB,SAAZB,C;MADoB,Y;K;wDAIrC,iB;MACI,oC  
AAa,mCAA0B,SAA1B,C;MAD6B,Y;K;;IAI9C,+B;MAAA,mD;MAG8B,sB9BRa,E8BQb,C;MAH9B,Y;K;IAKA,  
kD;MAAA,mD;MAIKD,sB9BdP,E8BcO,C;MAJID,Y;K;IAMA,2C;MAAA,mD;MAGqD,sBIENa,YkEMR,QIENQ,  
CkEMb,C;MAHrD,Y;K;ICrBJ,0C;MACI,IAAI,6BAAJ,C;QACU,KAAY,MAAK,UAAAL,C;;QAEIB,UAAU,KA AV  
,EAAwC,CAAxC,EA AiD,cAAN,KAAM,CAAjD,EAA4D,eAAW,UAA X,CAA5D,C;;K;IAMiB,kD;MAAA,uB;QA  
AgB,OAAA,kBAAW,SAAQ,CAAR,EA AW,CAAX,C;O;K;IAFPD,4C;MACI,IAAI,6BAAJ,C;QACI,iBAAiB,gC;Q  
ACX,KAAY,MAAK,UAAAL,C;;QAEIB,UAAU,KA AV,EAAwC,CAAxC,EA AiD,cAAN,KAAM,CAAjD,EAA4D,  
UAA5D,C;;K;IAIR,gE;MACI,IAAI,aAAY,UAAU,CAAV,IAAZ,CAAJ,C;QACI,UAAU,KA AV,EAAwC,SAAxC,  
EAAmD,UAAU,CAAV,IAAnD,EAAGe,UAAhE,C;Q;IAMiB,gC;MAAGB,OAAE,iBAAF,CAAE,EAAU,CAAV,C;  
K;IAF3C,0B;MACI,IAAI,6BAAJ,C;QACI,iBAAiB,gB;QACX,KAAY,MAAK,UAAAL,C;;QAEIB,UAAU,KA AV,E  
AAwC,CAAxC,EA AiD,cAAN,KAAM,CAAjD,EAA4D,cAA5D,C;;K;IAaa,kD;MAAoB,QAAC,IAAM,CAAP,KA  
Aa,IAAM,CAAnB,K;K;IARzC,uC;MACI,sC;QAAiC,OAAjC,yB;OACA,4BAA4B,K;MAE5B,YAAY,E;MAGZ,iB  
AAc,CAAd,UAA sB,GAAtB,U;QAAiC,KAAY,MAAK,KAAL,C;MAC7C,iBAAiB,kC;MACX,KAAY,MAAK,UA  
AL,C;MACIB,mBAAc,CAAd,YAAsB,KAAM,OAA5B,Y;QACI,QAAQ,MAAM,UAAQ,CAAR,IAAN,C;QACR,Q  
AAQ,MAAM,OAAN,C;QACR,IAAI,CAAC,IAAM,CAAP,OAAc,IAAM,CAApB,KAA0B,KA AK,CAAnC,C;UA

AsC,OAAO,K;;MAEjD,4BAA4B,I;MAC5B,OAAO,I;K;IAIX,2D;MACI,aAAa,gBAAmB,KAAM,OAAzB,O;MAC  
b,aAAa,YAAU,KAAY,EAAiB,MAAjB,EAAYB,KAAzB,EAAGC,YAAhC,EAA8C,UAA9C,C;MACb,IAAI,WAA  
W,KAAf,C;QACI,aAAU,KAAY,OAAiB,YAAjB,M;UAA+B,MAAM,CAAN,IAAW,OAAO,CAAP,C;Q;IAIID,4D;  
MAEI,IAAI,UAA5,GAAb,C;QACI,OAAO,K;OAGX,aAAa,CAAC,QAAQ,GAAR,IAAD,IAAgB,CAAhB,I;MACb  
,WAAW,YAAU,KAAY,EAAiB,MAAjB,EAAYB,KAAzB,EAAGC,MAAhC,EAAwC,UAAxC,C;MACX,YAAAY,Y  
AAU,KAAY,EAAiB,MAAjB,EAAYB,SAAS,CAAT,IAAZB,EAAGC,GAAR,C,EAA0C,UAA1C,C;MAEZ,aAAiB,S  
AAS,MAAb,GAAqB,KAArB,GAAgC,M;MAG7C,gBAAGB,K;MACHb,iBAAiB,SAAS,CAAT,I;MACjB,aAAU,K  
AAV,OAAiB,GAAjB,M;QAEQ,iBAAa,MAAb,IAAuB,cAAc,GAAR,C;UACI,gBAAGB,KAAK,SAAL,C;UACHb  
,iBAAiB,MAAM,UAAN,C;UAEjB,IAAI,UAAW,SAAQ,SAAR,EAAMB,UAANB,CAAX,IAA6C,CAAjD,C;YACI  
,OAAO,CAAP,IAAY,S;YACZ,6B;;YAEA,OAAO,CAAP,IAAY,U;YACZ,+B;;eAGR,iBAAa,MAAb,C;UACI,OA  
AO,CAAP,IAAY,KAAK,SAAL,C;UACZ,6B;;UAGA,OAAO,CAAP,IAAY,MAAM,UAAN,C;UACZ,+B;;MAMZ  
,OAAO,M;K;ICrGX,4C;MAMoB,UACM,M;MAHtB,IAAI,iBAAJ,C;QAAkB,OAAO,C;MACzB,aAAa,C;MACb,  
wBAAGB,SAAhB,gB;QAAgB,cAAA,SAAhB,M;QAEQ,oB;UAAmB,U;;UACnB,I1BFiC,MAAA,Y0BEnC,O1BFm  
C,C0BE9C,C;YAAwD,iCAAhC,OAAgC,C;iBAExD,uC;YAAmC,2BAAR,OAAQ,C;eACnC,wC;YAAmC,2BAAR  
,OAAQ,C;eACnC,sC;YAAmC,2BAAR,OAAQ,C;eACnC,uC;YAAmC,2BAAR,OAAQ,C;;YAEA,kBAAR,OAAQ,  
C;;QATvC,wB;QAYA,SAAS,MAAK,MAAL,QAAC,WAAd,I;;MAEb,OAAO,M;K;;ICTP,uC;MAAA,2C;K;2DAC  
I,0B;MAA2D,sBAAU,MAAV,C;K;gEAE3D,iB;MAA6C,Q;MAAA,wEAAqB,C;K;;IAHtE,mD;MAAA,kD;QAA  
A,iC;OAAA,2C;K;;MC0BA,iC;MAKA,8B;MA6CA,0BAAMe,I;;IAzEnE,kC;MAAA,oB;MAA+B,8C;K;2CAE3B,  
mB;MAAYD,MAAM,qCAA8B,iCAA9B,C;K;uCAC/D,Y;MACI,WAAa,Q;K;uDAGjB,mB;MAAgE,OAAA,WAAa  
,uBAAc,OAAAd,C;K;0CAE7E,Y;MAAwE,OAAA,iCAAY,W;K;qDAEpF,mB;MACI,IAAI,iBAAS,OAAT,CAAJ,C;  
QACI,WAAa,cAAO,OAAQ,IAAf,C;QACb,OAAO,I;OAEX,OAAO,K;K;wFAGY,Y;MAAQ,OAAA,WAAa,K;K;;  
8BA6ChD,Y;MACI,0BAAY,Q;K;0CAIhB,e;MAAMd,OAAA,0BAAY,gBAAS,GAAT,C;K;4CAE/D,iB;MAAmE,  
gBAAZ,0B;MAAY,c;;QvE+mDnD,Q;QADhB,IAAI,wCAAsB,mBAA1B,C;UAAqC,aAAO,K;UAAP,e;SACrB,2B;  
QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UAAAM,IuE/mDmD,uBAAS,gBvE+mD9C,OuE/mDwD,MAAV,QvE+mD  
5D,C;YAAwB,aAAO,I;YAAP,e;;QAC9C,aAAO,K;;MuEhDgD,iB;K;kFAInD,Y;MACI,IAAI,+BAAJ,C;QACI,0  
BAAW,qB;OAEf,OAAO,sC;K;uCAGf,Y;MAAgF,iC;K;kCAEHf,e;MAA+C,OAAA,0BAAY,WAAI,GAAJ,C;K;o  
CAE3D,sB;MAAgD,OAAA,0BAAY,aAAI,GAAJ,EAAS,KAAT,C;K;qCAE5D,e;MAAYC,OAAA,0BAAY,cAAO,  
GAAP,C;K;+EAEvB,Y;MAAQ,OAAA,0BAAY,K;K;;IA5DID,0C;MAAA,iD;MAAuD,8B;MAvC3D,mB;MAwCQ  
,8BAAMB,W;MACnB,2BAAGB,WAAAY,S;MAFhC,Y;K;IAKA,+B;MAAA,iD;MAGuB,aAAK,kEAAAL,Q;MAHvB  
,Y;K;IAKA,4D;MAAA,iD;MAQ8D,qB;M7EpC9D,IAAI,E6EsCQ,mBAAMB,C7EtC3B,CAAJ,C;QACI,c6EqCgC,  
+C;Q7EpChC,MAAM,gCAAYB,OAAQ,WAAjC,C;OAFV,IAAI,E6EuCQ,cAAc,C7EvCtB,CAAJ,C;QACI,gB6EsC  
2B,yC;Q7ErC3B,MAAM,gCAAYB,SAAQ,WAAjC,C;O6E0BV,Y;K;IAcA,gD;MAAA,iD;MAA2C,eAAK,eAAL,E  
AAsB,GAAtB,Q;MAA3C,Y;K;IAGA,yC;MAAA,iD;MAG8C,qB;MAC1C,KAAK,gBAAO,QAAP,C;MAJT,Y;K;I  
AqCJ,4B;MAK8E,gBAANe,aAAmB,gEAAAnB,C;MAA2E,wB;MAAIF,O1EvCO,S;K;;M2EjEP,uB;;kCAyCA,mB;  
MACI,UAAU,gBAAI,aAAI,OAAJ,EAAa,IAAb,C;MACd,OAAO,W;K;8BAGX,Y;MACI,gBAAI,Q;K;uCAOR,mB  
;MAA6D,OAAA,gBAAI,mBAAY,OAAZ,C;K;gCAEjE,Y;MAAYC,OAAA,gBAAI,U;K;iCAE7C,Y;MAAqD,OAA  
A,gBAAI,KAAK,W;K;qCAE9D,mB;MAAkD,OAAA,gBAAI,cAAO,OAAP,CAAJ,Q;K;+EAEpB,Y;MAAQ,OAA  
A,gBAAI,K;K;;IA5D1C,6B;MAAA,iD;MAGoB,8B;MAZxB,mB;MAAQ,oBAAM,gB;MAJV,Y;K;IAOA,yC;MAA  
A,iD;MAG2C,8B;MANB/C,mB;MAoBQ,oBAAM,eAAgB,QAAS,KAAzB,C;MACN,qBAAO,QAAP,C;MALJ,Y;K  
;IAQA,4D;MAAA,iD;MAQ2D,8B;MAhC/D,mB;MAiCQ,oBAAM,eAAgB,eAAhB,EAAiC,UAAjC,C;MATV,Y;K;  
IAYA,gD;MAAA,iD;MAA2C,eAAK,eAAL,EAAsB,GAAtB,Q;MAA3C,Y;K;IAEA,oC;MAAA,iD;MAM0C,8B;M  
A5C9C,mB;MA6CQ,oBAAW,G;MAPf,Y;K;IAMCJ,+B;MAKuC,gBAA5B,eAAQ,eAAR,C;MAAoC,6B;MAA3C,  
O3ENO,S;K;I4EzD6B,uC;MAAC,kC;MAErC,oBAAkC,kB;MACiC,sBAAYB,C;K;2EAHY,Y;MAAA,8B;K;2FAG  
rC,Y;MAAA,0B;K,OAAA,gB;MAAA,0B;K;gDAGA,sB;MACI,eAAe,aAAS,qBAAAY,GAAZ,C;MACxB,mBAAM  
B,6BAAsB,QAAtB,C;MACnB,IAAI,oBAAJ,C;QAEI,kBAAW,QAAX,IAAuB,mCAAY,GAAZ,EAAiB,KAAjB,C;  
;QAEvB,IAAI,6BAAJ,C;UAEI,YAA+B,Y;UAC/B,IAAI,aAAS,gBAAO,KAAM,IAAb,EAakB,GAAlB,CAAb,C;Y  
ACI,OAAO,KAAM,gBAAS,KAAT,C;;YAEb,kBAAW,QAAX,IAAuB,CAAQ,KAAR,EAae,mCAAY,GAAZ,EA  
AiB,KAAjB,CAAf,C;YACvB,6B;YACA,OAAO,I;;UAIX,YAAuC,Y;UACvC,cAAkB,wBAAN,KAAM,EAAiB,G

AAjB,C;UACIB,IAAI,eAAJ,C;YACI,OAAO,OAAM,gBAAS,KAAT,C;WAEX,KAAy,MAAK,mCAAY,GAAZ,E  
AAiB,KAAjB,CAAL,C;;;MAG1B,6B;MAEA,OAAO,I;K;iDAGX,e;MAEuB,Q;MADnB,eAAe,aAAS,qBAAY,GA  
AZ,C;MACL,oCAAsB,QAAtB,C;MAAA,iB;QAAMc,OAAO,I;OAA7D,mBAAMb,I;MACnB,IAAI,6BAAJ,C;QA  
CI,YAAgC,Y;QAChC,IAAI,aAAS,gBAAO,KAAM,IAAb,EAakB,GAAIB,CAAb,C;U5BzDR,O4B0D6B,iB5B1D  
vB,C4B0DmC,Q5B1DnC,C;U4B2DM,6B;UACA,OAAO,KAAM,M;;UAEb,OAAO,I;;;QAGX,YAAuC,Y;QACvC,  
8BAAc,KAAAd,iB;UACI,cAAY,MAAM,KAAN,C;UACZ,IAAI,aAAS,gBAAO,GAAP,EAAY,OAAM,IAAIB,CAA  
b,C;YACI,IAAI,KAAM,OAAN,KAAc,CAAIB,C;cACU,KAAN,UAA2B,C;c5BtE/C,O4BwEqC,iB5BxE/B,C4BwE  
2C,Q5BxE3C,C;;c4B2EoB,KAAy,QAAO,KAAP,EAAC,CAAd,C;;YAEtB,6B;YAEA,OAAO,OAAM,M;;;MAIzB,  
OAAO,I;K;0CAGX,Y;MACI,oBAAa,kB;MACb,YAAO,C;K;mDAGX,e;MAAyC,uBAAS,GAAT,S;K;8CAEzC,e;  
MAA+B,Q;MAAA,+BAAS,GAAT,8B;K;+CAE/B,e;MACuB,Q;MAAA,oCAAsB,aAAS,qBAAY,GAAZ,CAA/B,C  
;MAAA,iB;QAAoD,OAAO,I;OAA9E,mBAAMb,I;MACnB,IAAI,6BAAJ,C;QACI,YAAgC,Y;QAChC,IAAI,aAAS  
,gBAAO,KAAM,IAAb,EAakB,GAAIB,CAAb,C;UACI,OAAO,K;;UAEP,OAAO,I;;;QAGX,YAAuC,Y;QACvC,O  
AAa,wBAAN,KAAM,EAaiB,GAAjB,C;;K;uDAlrB,0B;MACI,sB;;Q7F+nCY,Q;QAAhB,iD;UAAgB,cAAhB,e;U  
AAsB,I6F/nCK,aAAS,gB7F+nCA,O6F/nCa,IAAb,M7F+nCd,C;YAAwB,qBAAO,O;YAAP,uB;;QAC9C,qBAAO,I;  
;M6FhoCH,yB;K;IAIO,8E;MAAA,wD;MACH,aAAY,E;MAEZ,YAA0B,MAAa,MAAK,qCAAL,C;MACvC,gBA  
Ae,E;MAEf,oBAA4B,I;MAC5B,eAAc,K;MACd,iBAAgB,E;MACHb,iBAAqC,I;K;yEAErC,Y;MACI,IAAI,6BAA  
wB,YAA5B,C;QACI,gBAAqB,iBAAqD,O;QAC1E,IAAI,4DAAc,SAaIB,C;UACI,OAAO,C;OAGf,IAAI,yDAAa,  
SAAK,QAAtB,C;QACI,oBAAe,2CAAW,UAAK,aAAL,CAAX,C;QACf,eAAU,iC;QACV,iBAAy,C;QACZ,OAA  
O,C;;QAEP,oBAAe,I;QACf,OAAO,C;;K;mEAI,f;Y;MACI,IAAI,eAAS,EAAb,C;QACI,aAAQ,oB;MACZ,OAAO,e  
AAS,C;K;gEAGpB,Y;MAEoB,Q;MADhB,IAAI,CAAC,cAAL,C;QAAGB,MAAM,6B;MACN,IAAI,YAAJ,C;QAC  
Z,yBAAqD,cAArD,C;;QAEa,OAAb,iB;;MAHJ,oB;MAKA,iBAAiB,S;MACjB,aAAQ,E;MACR,OAAO,S;K;kEAG  
X,Y;M/E/CR,I+EgDyB,c/EhDrB,QA AJ,C;QACI,cAhByB,0B;QAiBzB,MAAM,6BAAsB,OAAQ,WAA9B,C;O+E+  
CE,6BAAYB,cAAO,6BAAY,IAAnB,C;MACzB,iBAAy,I;MAEZ,uC;K;;6CatDZ,Y;MAEI,2D;K;4DAyDJ,oB;MA  
CI,mBAAMb,kBAAW,QAAX,C;MACnB,OAAW,iBAAiB,SAArB,GAAGC,IAAhC,GAA0C,Y;K;;;wCCtKrD,Y;  
MACI,aAAR,MAAM,OAAe,CAAP,IAAO,C;MAEb,OAAO,KAAP,IAAGB,C;M7BXpB,O6BYqB,M7BZf,C6BYu  
B,K7BZvB,C;M6BaF,OAAO,M;K;;ICNuB,qC;MAAC,kC;MAEnC,oBAAkC,kB;MACIC,sBAAYB,C;K;yEAHU,Y  
;MAAA,8B;K;yFAGnC,Y;MAAA,0B;K,OAAA,gB;MAAA,0B;K;iDAWA,e;MACI,IAAI,0BAAJ,C;QAAoB,OAA  
O,K;MAC3B,OAAO,kBAAW,GAAX,MAAoB,S;K;4CAG/B,e;MACI,IAAI,0BAAJ,C;QAAoB,OAAO,I;MAC3B,  
YAAy,kBAAW,GAAX,C;MACZ,OAAW,UAAU,SAArB,GAAGC,KAAhC,GAA2D,I;K;8CAI/D,sB;MjFVA,IAAI  
,EiFWQ,uBjFXR,CAAJ,C;QACI,cAda,qB;QAeb,MAAM,gCAAYB,OAAQ,WAAjC,C;OiFUN,eAAe,kBAAW,GA  
AX,C;MACf,kBAAW,GAAX,IAAkB,K;MAEIB,IAAI,aAAa,SAAjB,C;QACI,6B;QAEA,OAAO,I;;QAGP,OAAO,  
Q;;K;+CAIf,e;MACI,IAAI,0BAAJ,C;QAAoB,OAAO,I;MAC3B,YAAy,kBAAW,GAAX,C;MACZ,IAAI,UAAU,S  
AAAd,C;Q9BnDJ,O8BoDyB,iB9BpDnB,C8BoD+B,G9BpD/B,C;Q8BqDE,6B;QAEA,OAAO,K;;QAGP,OAAO,I;;K  
;wCAkf,Y;MACI,oBAAa,kB;MACb,YAAO,C;K;IAKA,0E;MAAA,oD;MACH,cAAkC,MAAa,MAAK,mCAAL,C  
;MAC/C,kBAA4B,qBAAL,WAAC,C;MAC5B,iBAA+B,I;K;iEAE/B,Y;MAAKC,OAAA,eAAS,U;K;8DAE3C,Y;M  
AIuB,gB;MAHnB,UAAU,eAAS,O;MACnB,iBAAU,G;MAES,+E;MAAnB,OAAO,iD;K;gEAGX,Y;MAEKc,UAA  
9B,M;MAAA,oC;MAA8B,YAAa,c;MjFchD,uB;MAeP,IAfoB,KAehB,QA AJ,C;QACI,cAhByB,0B;QAiBzB,MAA  
M,6BAAsB,OAAQ,WAA9B,C;;QAEN,sBAnBgB,K;;MiFde,oBAAO,sFAAP,C;K;;2CAjBnC,Y;MACI,yD;K;IAqB  
kD,0F;MAAA,8B;MAAA,oD;K;kHAC9B,Y;MAAQ,uB;K;oHACN,Y;MAAQ,6CAAuB,gBAAvB,C;K;2EAE9B,o  
B;MAAwC,OAAA,2BAAuB,aAAI,gBAAJ,EAAS,QAAT,C;K;qEAE/D,Y;MAA+B,OAAA,mCAAY,uBAAc,IAAd  
,C;K;qEAC3C,Y;MAAKC,OAAA,mCAAY,uBAAc,IAAd,C;K;mEAC9C,iB;MAA4C,OAAA,mCAAY,qBAAY,IA  
AZ,EAakB,KAAIB,C;K;;gDAR5D,e;MAAsD,iE;K;;;MCItD,sBAOsC,I;MA6CtC,yB;MAOA,4BAAkC,K;;IArIE,s  
D;MAZpC,oB;MAYyD,0CAAqC,GAARc,EAA0C,KAA1C,C;MACrD,oBAAuC,I;MACvC,oBAAuC,I;K;wDAEv  
C,oB;MACI,WAAMb,iB;MACnB,OAAa,mEAAS,QAAT,C;K;;IAIrB,wC;MAAA,oB;MAA+B,8C;K;IAE3B,sD;M  
AAA,oB;MACI,cACsC,I;MAEtC,cACsC,I;MAGIC,cAAO,iC;K;6DAIX,Y;MACI,OAAO,gBAAS,I;K;0DAGpB,Y;  
MAEI,IAAI,CAAC,cAAL,C;QAAGB,MAAM,6B;MAEtB,cAAc,0B;MACd,cAAO,O;MACa,gBAAb,OAAQ,a;;M  
AAf,c/E0DS,S+E1DoB,KAAO,iC/E0DzC,GAAqB,SAArB,GAA+B,I;M+EzD1B,OAAO,O;K;4DAGX,Y;MIFwBR  
,IAAI,EkFvBc,eAAQ,IIFuBtB,CAAJ,C;QACI,cAdW,e;QAeX,MAAM,6BAAsB,OAAQ,WAA9B,C;OkFxBE,WA

Ac,iB;MAGP,oCAAP,0BAAO,C;MACP,gCAAI,cAAO,0BAAO,IAAd,C;MAEJ,cAAO,I;K;;iDAIf,mB;MAAyD,M  
AAM,qCAA8B,iCAA9B,C;K;6CAC/D,Y;MACI,WAAmB,Q;K;6DAGvB,mB;MAAgE,OAAA,WAAmB,uBAAC,  
OAAAd,C;K;gDAEnF,Y;MAAwE,qD;K;2DAExE,mB;MACI,qB;MACA,IAAI,iBAAS,OAAT,CAAJ,C;QACI,WAA  
mB,cAAO,OAAQ,IAAf,C;QACnB,OAAO,I;OAEX,OAAO,K;K;8FAGY,Y;MAAQ,OAAA,WAAmB,K;K;sDAEI  
D,Y;MAAsC,WAAmB,iB;K;;iDAa7D,qB;MIFrBA,IAAI,EkF0BM,0BAAQ,IAAR,IAAgB,0BAAQ,IIF1B9B,CAAJ  
,C;QACI,cAdW,e;QAeX,MAAM,6BAAsB,OAAQ,WAA9B,C;OkF0BN,YAAy,mB;MACZ,IAAI,SAAS,IAAb,C;  
QACI,sBAAO,S;QACP,yBAAO,S;QACP,yBAAO,S;;QAGK,YAAa,KAAM,a;QIFIBhC,uB;QAeP,IAfoB,KAehB,  
QAAJ,C;UACI,gBAhByB,0B;UAIbZB,MAAM,6BAAsB,SAAQ,WAA9B,C;;UAEN,sBAnBgB,K;;QkFkBZ,+B;Q  
AEA,yBAAO,K;QACP,yBAAO,K;QAEP,qBAAa,S;QACb,qBAAa,S;;K;+CAIrB,qB;MAII,IAAI,SAAK,aAAL,KA  
Ac,SAAIb,C;QAEI,sBAAO,I;;QAEP,IAAI,wBAAS,SAAb,C;UAEI,sBAAO,sB;SAEX,qDAAc,sB;QACd,qDAAc,s  
B;;MAEIB,yBAAO,I;MACP,yBAAO,I;K;oCA8CX,Y;MAEI,qB;MACA,4BAAa,I;MACb,OAAO,I;K;oCAGX,Y;  
MACI,qB;MACA,kBAAI,Q;MACJ,sBAAO,I;K;gDASX,e;MAAmD,OAAA,kBAAI,mBAAY,GAAZ,C;K;kDAEv  
D,iB;MACiC,Q;MAAA,0B;MAAA,iB;QAAQ,OAAO,K;OAA5C,WAA6B,I;;QAEzB,IAAI,OAAA,IAAK,MAAL,  
EAAC,KAAAd,CAAJ,C;UACI,OAAO,I;SAEX,OAAO,cAAA,IAAK,aAAL,C;;MACF,iBAAS,mBAAT,C;MACT,O  
AAO,K;K;6CAIX,Y;MAAoF,uC;K;wCAEpF,e;MAAmD,Q;MAAJ,QAAl,OAAl,kBAAl,WAAI,GAAJ,CAAJ,6B;  
K;0CAE/C,sB;MACI,qB;MAEA,UAAU,kBAAl,WAAI,GAAJ,C;MACd,IAAI,OAAO,IAAX,C;QACI,eAAe,mCA  
AW,GAAx,EAAGB,KAAhB,C;QACf,kBAAl,aAAl,GAAJ,EAAS,QAAT,C;QACK,wBAAT,QAAS,C;QACT,OA  
AO,I;;QAEP,OAAO,GAAI,gBAAS,KAAT,C;;K;2CAInB,e;MACI,qB;MAEA,YAAy,kBAAl,cAAO,GAAP,C;MA  
ChB,IAAI,SAAS,IAAb,C;QACU,sBAAN,KAAM,C;QACN,OAAO,KAAM,M;OAEjB,OAAO,I;K;qFAGmB,Y;M  
AAQ,OAAA,kBAAl,K;K;6CAE1C,Y;MACI,IAAI,yBAAJ,C;QAAGB,MAAM,oC;K;;IANg1B,mC;MAAA,uD;MA  
GuB,qB;MA9J3B,yB;MA+JQ,sBAAM,gB;MAJV,Y;K;IAOA,iD;MAAA,uD;MAAoD,qB;MAIKxD,yB;MAoKc,Q;  
MAAN,sBAAM,+D;MAFV,Y;K;IAKA,kE;MAAA,uD;MAQ8D,eAAM,eAAN,EAAuB,UAAvB,Q;MA/KIE,yB;M  
AgLQ,sBAAM,gB;MATV,Y;K;IAYA,sD;MAAA,uD;MAA2C,qBAAK,eAAL,EAASB,GAAtB,Q;MAA3C,Y;K;IA  
EA,+C;MAAA,uD;MAG2C,qB;MAxL/C,yB;MAyLQ,sBAAM,gB;MACN,KAAK,gBAAO,QAAP,C;MALT,Y;K;I  
A6EJ,kC;MAKwD,gBAA7C,qBAAYB,eAAzB,C;MAAqD,wB;MAA5D,O/EjMO,S;K;;oCgFvCP,Y;MAEK,Q;MA  
A8B,CAA9B,2EAA8B,S;MAC/B,OAAO,I;K;6CAGX,Y;MAA+C,gBAAl,iB;K;;IAhCnD,wC;MAAA,uD;MAAmD  
,eAAM,GAAN,Q;MAPvD,yB;MAOI,Y;K;IAEA,qC;MAAA,uD;MAGuB,eAAM,oBAAN,Q;MAZ3B,yB;MASI,Y;  
K;IAKA,+C;MAAA,uD;MAG8C,eAAM,oBAAN,Q;MAJBID,yB;MAkBQ,qBAAO,QAAP,C;MAJJ,Y;K;IAOA,kE;  
MAAA,uD;MAQ8D,eAAM,qBAAsB,eAAtB,EAAuC,UAAvC,CAAN,Q;MA7BIE,yB;MAqBI,Y;K;IAUA,sD;MA  
AA,uD;MAA2C,qBAAK,eAAL,EAASB,GAAtB,Q;MAA3C,Y;K;IAGBJ,qC;MAKMD,gBAAXC,mBAAC,qBAAd,C  
;MAAGD,6B;MAAvD,OhFoBO,S;K;;kFiFzEX,uB;MAQI,OAAO,O;K;ICXX,sB;K;mCACI,Y;MACI,mBAAM,IA  
AN,C;K;2CAGJ,mB;MACI,mBAAM,OAAN,C;MACA,c;K;iCAKJ,Y;K;;IAKuB,oC;MAA8B,qB;MAA7B,gC;K;2  
CACxB,mB;MAEI,oBA+DyC,OA/Dd,OA+Dc,C;MA9DzC,iBAAa,OAAM,aAAN,C;K;;IAIrB,8B;MAEoC,qB;K;i  
DACHc,mB;MACI,OAAQ,KAAI,OAAJ,C;K;mDAGZ,mB;MACI,OAAQ,KAAI,OAAJ,C;K;2CAGZ,Y;MACI,OA  
AQ,KAAI,EAAJ,C;K;;IAIhB,0B;MAEQC,qB;MACjC,cAAa,E;K;6CAEb,mB;MACI,eAoCyC,OApCxB,OAoCwB,  
C;K;qCAjC7C,Y;MACI,cAAS,E;K;;IAIjB,sC;MAE4C,yB;K;yDACxC,mB;MACI,QAwByC,OAxB1B,OAwb0B,C  
;MAvBzC,QAAQ,CxEqJoF,awErJhE,IxEqJgE,EwErJ1D,CxEqJ0D,C;MwEpJ5F,IAAI,KAAC,CAAT,C;QACI,4BA  
AU,CxE+J0E,WwE/J9D,CxE+J8D,EwE/J3D,CxE+J2D,C;QwE9JpF,Y;QACA,IAAI,CxE0JiE,WwE1JrD,IAAI,CA  
AJ,IxE0JqD,C;OwExJzE,4BAAU,C;K;iDAGd,Y;MACI,OAAQ,KAAI,WAAJ,C;MACR,cAAS,E;K;;IAWjB,yB;M  
ACiD,cAAa,KAAb,C;K;IAEjD,mB;MAEI,MAAO,U;K;IAGX,4B;MAEI,MAAO,iBAAQ,OAAR,C;K;IAGX,wB;  
MAEI,MAAO,eAAM,OAAN,C;K;IAGX,kB;MACqC,MAAM,qCAA8B,sCAA9B,C;K;IAE3C,wB;MAC4C,MAA  
M,qCAA8B,4CAA9B,C;K;ICIGID,mD;MACI,0B;MASA,gBAA2B,a;K;2FAFvB,Y;MAAQ,OAAA,eAAS,Q;K;oD  
AIrB,kB;MACI,UAAU,IAAK,S;MAEX,YAAQ,2CAAR,C;QACI,gBAAC,MAAO,M;WAEzB,YAAQ,yBAAR,C;Q  
ACI,gBAAC,yC;QACd,eAAS,oBAAW,MAAX,C;;QAEI,MAAM,6BAAsB,iBAAtB,C;K;4CAItB,Y;MAOW,Q;M  
ALP,IAAI,kBAAW,2CAAf,C;QACI,gBAAS,yB;QACT,OAAO,yB;OAEX,aAAa,IAAK,S;MAEd,eAAW,yCAAX,  
C;QAASB,gC;WACTb,0C;QAA4B,MAAM,MAAO,U;;QACjC,a;MAHZ,W;K;;IA7BJ,gD;MAAA,0D;MACyD,6B  
AAK,QAAL,EAae,2CAAf,C;MADzD,Y;K;;;ICRA,2C;MAAA,+D;MAAuB,iC;MAF3B,iC;MAEI,Y;K;IACA,sD  
;MAAA,+D;MAAuC,6BAAM,OAAN,Q;MAH3C,iC;MAGI,Y;K;IACA,6D;MAAA,+D;MAAmD,kCAAM,OAAN,

EAAe,KAAf,C;MAJvD,iC;MAII,Y;K;IACA,oD;MAAA,+D;MAAiC,6BAAM,KAAN,Q;MALrC,iC;MAKI,Y;K;Ix  
C4CJ,yE;MASI,sC;MAAA,4C;K;IATJ,iGAWY,Y;MAAQ,2B;KAXpB,E;IAAA,0DAaQ,kB;MACI,wBAAW,MAA  
X,C;K;IAdZ,sF;IyC5C2E,0C;M1CkKhE,Q;MADP,e0ChKA,M1CgKA,C;MACO,Q0CjKP,M1CiKO,+D;M0ChKX,  
W;K;;+FCuHA,gB;MACI,aAAa,IAAb,MAAa,E;MACb,KAaK,MAAL,C;MACA,OAAO,M;K;wFC3HX,yB;MAA  
A,uD;MAAA,wC;QAWqG,OAAK,cAAL,SAaK,EAAiB,IAAjB,EAAuB,IAAvB,C;O;KAX1G,C;wFAaA,yB;MAA  
A,uD;MAAA,wC;QAWoG,OAAK,cAAL,SAaK,EAAiB,IAAjB,EAAuB,IAAvB,C;O;KAXzG,C;8ECbA,yB;MAA  
A,6C;MAAA,sC;QAOyD,OAAK,SAAL,SAaK,EAAy,QAaZ,C;O;KAP9D,C;8EASA,yB;MAAA,6C;MAAA,wC;  
QAWkE,OAAK,SAAL,SAaK,EAAa,UAAb,S;O;KAXvE,C;oFAaA,yB;MAAA,mD;MAAA,wC;QAWqE,OAAK,  
YAAL,SAaK,EAAgB,UAAhB,S;O;KAX1E,C;kFCZI,yB;MAAA,iD;MAAA,4B;QAAe,OAAK,WAAL,SAaK,C;  
O;KAApB,C;wFAYA,yB;MAAA,uD;MAAA,4B;QAAe,OAAK,cAAL,SAaK,C;O;KAApB,C;IC5BJ,gC;MAAoE,  
gCAAqB,OAArB,C;K;IAEIC,uC;MAAC,wB;K;iDAC/B,iB;MACI,eAAQ,KAAR,C;K;8CAGJ,Y;MAAyC,iCAAuB  
,cAAvB,M;K;;ICCO,6C;MAAA,8B;MAAS,uB;K;8FACIC,Y;MAAQ,OAAA,gBAAY,O;K;mDAE3C,iB;MACI,IA  
DoC,KACpC,IAAG,CAAH,IADoC,KACpC,IAAM,sBAAN,C;QAD8B,OACX,gBAAY,MAAK,KAAL,C;;QACvB  
,MAAM,8BAA0B,WAAQ,KAAR,6BAAmC,sBAAnC,MAA1B,C;K;;IARtB,8B;MAGoD,4C;K;wECFpD,yB;MA  
AA,uC;MAAA,4B;QAOsC,MAAL,SAaK,C;O;KAPtC,C;kFASA,yB;MAAA,iD;MAAA,kC;QAWuD,OAAK,WA  
AL,SAaK,EAAc,IAAd,C;O;KAX5D,C;+ECfA,qB;MAI8C,gB;K;iFAE9C,qB;MAIsE,OAAK,S;K;kFAE3E,qB;MA  
MyE,gB;K;IAEzE,6B;MAiBa,UAPF,M;MAFP,QAaC,S;MAGV,cAAK,UAAL,U;QACI,mBAAK,UAAL,G;;QACJ  
,I/CzBqC,MAAa,Y+CyBvC,C/CzBuC,C+CyBID,C;UAC6B,8BAAzB,CAAyB,C;;UAGN,UAAIB,uDAaKB,Y;;MA  
P3B,a;K;IC9BJ,2B;MAEI,MAAM,yBAAqB,OAArB,C;K;IAGV,sB;MAEI,MAAM,uBAAmB,cAAAnB,C;K;IAGV,2  
B;MAEI,MAAM,6BAAsB,OAAtB,C;K;IAGV,iC;MAEI,MAAM,4CAAqC,uBAAqB,YAArB,8BAArC,C;K;ICIBV  
,8B;MC8CW,kB1GqBiD,oB;M0GM9C,Q;MAAA,OAAK,0B;MAAf,OAAU,cAAV,C;QAAU,mB;QACN,UAAU,s  
BAAM,CAAN,C;QACV,kBAaKB,sBAAY,GAAZ,C;QAKFiD,U;QAJFnE,W1GuKJ,a0GvKgB,G1GuKhB,EyG1Oo  
B,CCmEkC,uBAAuB,CAAC,WAAy,mBAAY,GAAZ,CAiFhD,GDpJrC,CCoJqC,GAA6B,UAJfJc,WaifiC,6DDp  
JnD,IAAM,CAAN,IzG0OpB,C;;MyG1OA,OCqEO,W;K;;;ICjCX,qB;MAK0B,Q;MADtB,UAAmB,E;MACnB,wB  
AAsB,KAAtB,gB;QAAsB,aAAA,KAAtB,M;QAAK,IAAC,0BAAD,EAAO,2B;QACR,IAAI,IAAJ,IAAY,K;;MAE  
hB,OAAO,G;K;IAGX,+B;MAMgB,Q;MADZ,WAA0B,MAAa,MAAK,KAAL,C;MACvC,wBAAY,IAAZ,gB;QAA  
Y,UAAA,IAAZ,M;QACI,IAAU,KAAY,gBAaE,GAaf,CAAtB,C;UACI,UAAK,GAAL,IAAY,MAAM,GAAN,C;;  
MAGpB,OAAO,S;K;qEC5DX,yB;MAAA,iB;MAAA,oB;QAOkD,OAAA,MAAW,KAAI,CAAJ,C;O;KAP7D,C;qE  
ASA,yB;MAAA,iB;MAAA,oB;QAOkD,OAAA,MAAW,KAAI,CAAJ,C;O;KAP7D,C;qEASA,yB;MAAA,iB;MA  
AA,oB;QAOkD,OAAA,MAAW,KAAI,CAAJ,C;O;KAP7D,C;uEASA,yB;MAAA,iB;MAAA,oB;QASmD,OAAA,  
MAAW,MAAK,CAAL,C;O;KAT9D,C;uEAWA,yB;MAAA,iB;MAAA,oB;QASmD,OAAA,MAAW,MAAK,CAA  
L,C;O;KAT9D,C;uEAWA,yB;MAAA,iB;MAAA,oB;QASmD,OAAA,MAAW,MAAK,CAAL,C;O;KAT9D,C;yEA  
WA,yB;MAAA,iB;MAAA,uB;QAKb+D,OAAA,MAAW,OAAM,CAAN,EAAS,CAAT,C;O;KAIB1E,C;uEAoBA,y  
B;MAAA,iB;MAAA,oB;QAUmD,OAAA,MAAW,MAAK,CAAL,C;O;KAV9D,C;uEAYA,yB;MAAA,iB;MAAA,  
oB;QASmD,OAAA,MAAW,MAAK,CAAL,C;O;KAT9D,C;uEAWA,yB;MAAA,iB;MAAA,oB;QAUmD,OAAA,  
MAAW,MAAK,CAAL,C;O;KAV9D,C;yEAYA,yB;MAAA,iB;MAAA,oB;QAYoD,OAAA,MAAW,OAAM,CAA  
N,C;O;KAZ/D,C;yEAca,yB;MAAA,iB;MAAA,oB;QAYoD,OAAA,MAAW,OAAM,CAAN,C;O;KAZ/D,C;yEAca  
A,yB;MAAA,iB;MAAA,oB;QAaoD,OAAA,MAAW,OAAM,CAAN,C;O;KAb/D,C;yEAeA,yB;MAAA,iB;MAAA,  
uB;QAS+D,OAAA,MAAW,OAAM,CAAN,EAAS,CAAT,C;O;KAT1E,C;uEAWA,yB;MAAA,iB;MAAA,oB;QA  
QmD,OAAA,MAAW,MAAK,CAAL,C;O;KAR9D,C;qEAUA,yB;MAAA,iB;MAAA,oB;QAUkD,OAAA,MAAW,  
KAAI,CAAJ,C;O;KAV7D,C;yEAYA,yB;MAAA,iB;MAAA,oB;QAcO,D,OAAA,MAAW,OAAM,CAAN,C;O;KAd  
/D,C;IAGBA,sB;MAcI,IAAI,QAAQ,GAAR,IAAE,SAaQ,GAA3B,C;QAAGC,OAAO,wCAAo,I;MAC9C,OAAO,I  
AAW,KAAI,CAAJ,CAAX,GAAoB,IAAW,KAAI,IAAJ,C;K;mEAG1C,yB;MAAA,iB;MAAA,oB;QAWiD,OAAA,  
MAAW,KAAI,CAAJ,C;O;KAX5D,C;yEAaA,yB;MAAA,iB;MAAA,oB;QAOoD,OAAA,MAAW,OAAM,CAAN,  
C;O;KAP/D,C;uEASA,yB;MAAA,iB;MAAA,oB;QAOmD,OAAA,MAAW,MAAK,CAAL,C;O;KAP9D,C;uEASA  
,yB;MAAA,iB;MAAA,oB;QAgBmD,OAAA,MAAW,OAAM,CAAN,C;O;KAhB9D,C;uEakBA,yB;MAAA,iB;M  
AAA,oB;QAUmD,OAAA,MAAW,MAAK,CAAL,C;O;KAV9D,C;yEAYA,yB;MAAA,iB;MAAA,oB;QAUoD,OA  
AA,MAAW,OAAM,CAAN,C;O;KAV/D,C;+EAYA,yB;MAAA,iB;MAAA,oB;QAUuD,OAAA,MAAW,OAAM,C

AAN,C;O;KAVIE,C;IA YA,kB;MAQI,IAAI,IAAI,GAAJ,KAAW,GAAf,C;QACI,OAAO,IAAW,OAAM,CAAN,C;  
OAEtB,YAzBgD,MAAW,OAYBzC,CAzByC,C;MA0B3D,OAAW,QAAQ,CAAR,KAAa,GAAxB,GAA6B,KAA7B  
,GAtC+C,MAAW,MA sCb,CAtCa,C;K;qEAyC9D,yB;MAAA,iB;MAAA,oB;QAUkD,OAAA,MAAW,KAAI,CAAJ  
,C;O;KAV7D,C;uEAYA,yB;MAAA,iB;MAAA,oB;QAWmD,OAAA,MAAW,MAAK,CAAL,C;O;KAX9D,C;wEA  
cA,yB;MAAA,iB;MAAA,uB;QAO6D,OAAA,MAAW,KAAI,CAAJ,EAAO,CAAP,C;O;KAPxE,C;wEASA,yB;MA  
AA,iB;MAAA,uB;QAO6D,OAAA,MAAW,KAAI,CAAJ,EAAO,CAAP,C;O;KAPxE,C;qEAWA,yB;MAAA,iB;M  
AAA,+B;QAayD,OAAA,MAAW,KAAI,SAAJ,EAAU,CAAV,C;O;KAbpE,C;uEAeA,yB;MAAA,iB;MAAA,+B;Q  
AOsD,OAAA,MAAW,KAAI,SAAJ,EAAU,CAAZ,C;O;KAPjE,C;iGAmBsD,yB;MAAA,iB;MAAA,4B;QAAQ,OA  
AA,MAAW,KAAI,SAAJ,C;O;KAA nB,C;+EAaT,yB;MAAA,iB;MAAA,4B;QAAQ,OAAA,MAAW,MAAK,SAAL  
,C;O;KAA nB,C;fAE7C,yB;MAAA,6C;MAAA,kC;QAK8D,OAAK,SAAL,SAAK,EAAC,IAAd,C;O;KALnE,C;IA  
kBqC,4B;MACjC,gBAAO,CAAP,C;QADyC,OACrB,QAAP,CAAC,SAAM,C;WACpB,IAAK,QAAL,SAAK,CAA  
L,IAAgB,cAAQ,wCAAo,kBAA/B,C;QAFyC,OAEW,S;WACpD,kBAAQ,wCAAo,UAAf,C;QAHyC,OAGb,YAA  
Y,SAAL,SAAK,C;;QAHc,OAI5B,OAAL,SAAK,CAAL,GAAGB,S;K;IAG5B,2B;MAKI,IAAK,QAAL,SAAK,CA  
AL,IAAgB,cAAQ,wCAAo,kBAA/B,C;QADwC,OACY,S;WACpD,kBAAQ,GAAR,C;QAFwC,OAeZB,wCAAo,  
U;;QACP,WAAc,UAAAL,SAAK,CAAL,yBAAuB,YAAO,CAAX,GAAC,CAAd,GAAqB,EAAXC,E;QAHgB,OhDhb  
6B,MAAA,gBAAe,IAAf,C;;K;IgdSbtF,6B;MAKI,IAAK,QAAL,SAAK,CAAL,IAAgB,cAAQ,wCAAo,kBAA/B,C;  
QAD0C,OACU,S;WACpD,kBAAQ,GAAR,C;QAF0C,OAe3B,CAAC,wCAAo,U;;QACR,WAAc,UAAAL,SAAK,C  
AAL,yBAAuB,YAAO,CAAX,GAAC,EAAd,GAAsB,CAAZC,E;QAHkB,OhD1b2B,MAAA,gBAAe,IAAf,C;;K;Igd  
ctF,oC;MAUI,IAAK,QAAL,SAAK,CAAL,IAAmB,QA AH,EAAG,CAAnB,C;QADuD,OACzB,wCAAo,I;WACrC  
,WAAM,SAAN,C;QAFuD,OAeZC,E;WACd,SAAK,SAAL,C;QAHuD,OAGrC,OAAL,SAAK,C;;QAHqC,OAI1B,  
SAAL,SAAK,C;K;IAIjC,+B;MAYI,uB;QAAW,MAAM,gCAAYB,yBAAzB,C;WACjB,gBAAO,UAAp,C;QAFyC,  
OAEjB,U;WACxB,gBAAO,WAAp,C;QAHyC,OAGjB,W;;QAHiB,OAIv,YAAvB,IAAW,OAAM,SAAN,CAAY,  
C;K;IAGnC,gC;MAYI,uB;QAAW,MAAM,gCAAYB,yBAAzB,C;WACjB,oD;QAF2C,+B;WAG3C,oD;QAH2C,+B  
;;QAAA,OAIz,uBAAvB,IAAW,OAAM,SAAN,CAAY,C;K;uEASnC,yB;MAAA,iB;MAAA,oB;QAOgD,OAAA,  
MAA6B,KAAZ,CAAY,C;O;KAP7E,C;uEASA,yB;MAAA,iB;MAAA,oB;QAOgD,OAAA,MAA6B,KAAZ,CAAY  
,C;O;KAP7E,C;uEASA,yB;MAAA,iB;MAAA,oB;QAOgD,OAAA,MAA6B,KAAZ,CAAY,C;O;KAP7E,C;yEASA  
,yB;MAAA,iB;MAAA,oB;QASiD,OAAA,MAA8B,MAAZ,CAAY,C;O;KAT/E,C;yEAWA,yB;MAAA,iB;MAAA,  
oB;QASiD,OAAA,MAA8B,MAAZ,CAAY,C;O;KAT/E,C;yEAWA,yB;MAAA,iB;MAAA,oB;QASiD,OAAA,MA  
A8B,MAAZ,CAAY,C;O;KAT/E,C;2EAWA,yB;MAAA,iB;MAAA,uB;QAKB4D,OAAA,MAA6C,OAA1B,CAA0  
B,EA AZ,CAAY,C;O;KAIbZG,C;yEAoBA,yB;MAAA,iB;MAAA,oB;QAUiD,OAAA,MAA8B,MAAZ,CAAY,C;O  
;KAV/E,C;yEAYA,yB;MAAA,iB;MAAA,oB;QASiD,OAAA,MAA8B,MAAZ,CAAY,C;O;KAT/E,C;yEAWA,yB;  
MAAA,iB;MAAA,oB;QAUiD,OAAA,MAA8B,MAAZ,CAAY,C;O;KAV/E,C;2EAYA,yB;MAAA,iB;MAAA,oB;  
QAYkD,OAAA,MAA+B,OAAZ,CAAY,C;O;KAZjF,C;2EAca,yB;MAAA,iB;MAAA,oB;QAYkD,OAAA,MAA+  
B,OAAZ,CAAY,C;O;KAZjF,C;2EAca,yB;MAAA,iB;MAAA,oB;QAakD,OAAA,MAA+B,OAAZ,CAAY,C;O;K  
AbjF,C;2EAeA,yB;MAAA,iB;MAAA,uB;QAS4D,OAAA,MAA6C,OAA1B,CAA0B,EA AZ,CAAY,C;O;KATzG,C  
;yEAWA,yB;MAAA,iB;MAAA,oB;QAQiD,OAAA,MAA8B,MAAZ,CAAY,C;O;KAR/E,C;uEAUA,yB;MAAA,iB  
;MAAA,oB;QAUgD,OAAA,MAA6B,KAAZ,CAAY,C;O;KAV7E,C;2EAYA,yB;MAAA,iB;MAAA,oB;QAcKd,O  
AAA,MAA+B,OAAZ,CAAY,C;O;KAdjF,C;uEA gBA,yB;MAAA,mC;MAAA,0B;QAc6D,OAAmC,IAA7B,CAA6  
B,EA AZ,IAAY,C;O;KAdhG,C;qEA gBA,yB;MAAA,iB;MAAA,oB;QAW+C,OAAA,MAA6B,KAAZ,CAAY,C;O;  
KAX5E,C;2EAaA,yB;MAAA,iB;MAAA,oB;QAOkd,OAAA,MAA+B,OAAZ,CAAY,C;O;KAPjF,C;yEASA,yB;  
MAAA,iB;MAAA,oB;QAOiD,OAAA,MAA8B,MAAZ,CAAY,C;O;KAP/E,C;yEASA,yB;MAAA,iB;MAAA,oB;Q  
AgBiD,OAAA,MAA+B,OAAZ,CAAY,C;O;KAhBhF,C;yEAkBA,yB;MAAA,iB;MAAA,oB;QAUiD,OAAA,MAA  
8B,MAAZ,CAAY,C;O;KAV/E,C;2EAYA,yB;MAAA,iB;MAAA,oB;QAUkD,OAAA,MAA+B,OAAZ,CAAY,C;O;  
KAVjF,C;fAYAY,yB;MA3gBA,iB;MA2gBA,oB;QAUqD,OA3gBE,MAAW,OA2gBF,CA3gBE,C;O;KAigBIE,C;2  
EAYA,yB;MAAA,uC;MAAA,oB;QAQkD,OAAoB,MAAZ,CAAY,C;O;KARtE,C;uEAWA,yB;MAAA,iB;MAAA,  
oB;QAUgD,OAAA,MAA6B,KAAZ,CAAY,C;O;KAV7E,C;yEAYA,yB;MAAA,iB;MAAA,oB;QAWiD,OAAA,M  
AA8B,MAAZ,CAAY,C;O;KAX/E,C;wEAeA,yB;MAAA,iB;MAAA,uB;QAO0D,OAAA,MAAW,KAAI,CAAJ,EA  
AO,CAAP,C;O;KAPrE,C;wEASA,yB;MAAA,iB;MAAA,uB;QAO0D,OAAA,MAAW,KAAI,CAAJ,EAAO,CAAP,

C;O;KAPrE,C;sEAYA,yB;MAAA,iB;MAAA,+B;QAAsD,OAAA,MAA8C,KAA1B,SAA0B,EAAZ,CAAY,C;O;K  
AbpG,C;uEAeA,yB;MAAA,iB;MAAA,+B;QAOoD,OAAA,MAA8C,KAA1B,SAA0B,EAAZ,CAAY,C;O;KAPIG,  
C;kGAmBoD,yB;MAAA,iB;MAAA,4B;QAAQ,OAAA,MAAgC,KAAZ,SAAY,C;O;KAAxC,C;gFAaT,yB;MAAA  
,iB;MAAA,4B;QAAQ,OAAA,MAAiC,MAAZ,SAAY,C;O;KAAzC,C;gFAE3C,yB;MAAA,6C;MAAA,kC;QAO8D  
,OAA0C,SAArC,SAaQc,EAAZ,IAAY,C;O;KAPxG,C;iFASA,yB;MAAA,6C;MAAA,kC;QAK4D,OAA0C,SAArC  
,SAaQc,EAAZ,IAAY,C;O;KALtG,C;oFAQA,yB;MAAA,iD;MAAA,4B;QAYmD,OAAW,WAAZ,SAAW,C;O;K  
AZ9D,C;sFAcA,yB;MAAA,mD;MAAA,4B;QAYqD,OAAW,YAAX,SAAW,C;O;KAZhE,C;IAoBA,kB;MAUqC,  
OAAI,IAAI,CAAR,GAAY,CAAC,CAAD,OAAM,CAAIB,GAA0B,C;K;wEAE/D,yB;MAAA,iB;MAAA,uB;QAK  
oD,OAAA,MAAW,KAAI,CAAJ,EAAO,CAAP,C;O;KAL/D,C;wEAOA,yB;MAAA,iB;MAAA,uB;QAKoD,OAAA  
,MAAW,KAAI,CAAJ,EAAO,CAAP,C;O;KAL/D,C;mGaiBgD,yB;MAAA,mC;MAAA,4B;QAAQ,WAAI,SAAJ,C  
;O;KAAR,C;IAShB,+B;MAC5B,gBAAO,CAAP,C;QADoC,OACxB,E;WACZ,gBAAO,CAAP,C;QAFoC,OAExB,  
C;;QAFwB,OAG5B,C;K;IAKZ,kB;MASuC,OAAI,eAAI,CAAR,GAAY,CAAD,aAAX,GAAMb,C;K;wEAE1D,gB;  
MAKuD,OAAI,kBAAK,CAAL,MAAJ,GAAY,CAAZ,GAAMb,C;K;wEAE1E,gB;MAKuD,OAAI,kBAAK,CAAL,  
MAAJ,GAAY,CAAZ,GAAMb,C;K;mGAYxB,yB;MAAA,mC;MAAA,4B;QAAQ,WAAI,SAAJ,C;O;KAAR,C;IA  
SjB,+B;MAC7B,2BAAO,CAAP,C;QADqC,OACzB,E;WACZ,2BAAO,CAAP,C;QAFqC,OAExB,C;;QAFyB,OAG  
7B,C;K;IC1kCZ,4B;MAI4C,qBAAQ,S;K;IAEpD,4B;MAI2C,qBAAQ,S;K;IAEnD,+B;MAGiD,qBAAQ,wCAAO,k  
BAAf,IAAoC,cAAQ,wCAAO,kB;K;IAEpG,iC;MAGgD,qBAAQ,uCAAM,kBAAd,IAAmC,cAAQ,uCAAM,kB;K;I  
AEjG,6B;MAG+C,QAAC,qBAAD,IAAiB,CAAC,kB;K;IAEjE,+B;MAG8C,QAAC,uBAAD,IAAiB,CAAC,kB;K;I  
AGhE,iC;MAOI,QAAQ,S;MACR,IAAI,CAAC,IAAM,UAAP,KAAsB,CAAE,KAAK,CAAP,GAAC,UAAPc,K;M  
ACJ,IAAI,CAAC,IAAM,SAAP,KAAsB,CAAE,KAAK,CAAP,GAAC,SAAPc,K;MACJ,IAAI,CAAC,IAAM,SAAP  
,KAAsB,CAAE,KAAK,CAAP,GAAC,SAAPc,K;MACJ,IAAI,CAAC,IAAM,QAAP,KAAsB,CAAE,KAAK,CAAP,  
GAAC,QAAPc,K;MACJ,IAAI,CAAC,IAAM,KAAP,KAAsB,CAAE,KAAK,EAA7B,K;MACJ,OAAO,C;K;kGAG  
X,yB;MAAA,iB;MAAA,4B;QAM2D,OAAA,MAAO,OAAM,SAAN,C;O;KANIE,C;IAQA,0C;MAOI,YATuD,MA  
AO,OAS9B,EAAf,aAAQ,CAAC,SAAD,IAAR,CAAe,CAT8B,CAS9D,I;K;IAEJ,sC;MAOI,OAAI,cAAQ,CAAZ,G  
AAe,CAAf,GAAsB,CAAE,IAAI,EAAJ,GAIB+B,MAAO,iB;K;IAoBIE,qC;MAQI,oBAAS,CAAC,SAAD,IAAT,C;  
K;IAEJ,yC;MAaI,oBAAI,QAAJ,GAAiB,cAAK,EAAL,GAAqB,Q;K;IAG1C,0C;MAaI,oBAAI,EAAJ,GAAoB,QA  
ApB,GAAiC,cAAK,Q;K;IAG1C,mC;MAMI,OAAK,ajDhEmD,uBiDgEnD,CAAL,GAA0B,ajDjE6B,sBiDiE7B,CA  
A1B,I;K;IAEJ,2C;MAMU,WAAW,SjDxEuC,c;MiDyEpD,e;QADJ,OACS,KA7E8C,MAAO,OjDGP,sBiDHO,CA6  
ErD,I;;QADT,OA5EuD,MAAO,OA8EID,IA9EkD,C;;K;IAiFIE,4C;MAMU,UAAU,SjDpFuC,a;MiDqFnD,c;QADJ,  
OACS,KAAqB,sBjDpF0B,uBiDoF1B,CAArB,I;;QADT,OAEGb,sBAAJ,GAAI,C;K;IAGpB,wC;MAOU,WAAW,S  
jD/FuC,c;MiDgGpD,e;QAAK,UAAS,kBjDjGqC,sBiDiGrC,C;QADIB,OjDjG4C,MAAa,KAAK,UAAS,GAAT,EiD  
kGvB,CjDlGuB,C;;QiDmGID,aAAa,kBAAL,IAAK,C;QAFzB,OjDjG4C,MAAa,KAAK,UiDmG7C,CjDnG6C,EA  
Ac,MAAd,C;;K;liDsGIE,uC;MAOU,UAAU,SjD5GuC,a;MiD6GnD,c;QAAK,WAAa,iBjD5GkC,uBiD4GIC,C;QA  
DtB,OjD7G4C,MAAa,KAAK,UiD8GhD,CjD9GgD,EAAC,IAAd,C;;QiD+GID,YAAS,iBAAJ,GAAI,C;QAFrB,OjD  
7G4C,MAAa,KAAK,UAAS,KAAT,EiD+GrB,CjD/GqB,C;;K;liDkHIE,2C;MAaI,IAAI,CAAC,WAAa,EAAd,MAA  
qB,CAAZB,C;QACI,UAAU,SjD/HyC,a;QiDgInD,WAAW,SjD/HyC,c;QiDgIpD,aAAa,GAAL,IAAI,QAAR,GAAqB  
,IAAK,MAAK,CAAC,QAAD,IAAL,C;QACvC,cAAc,IAAK,IAAI,QAAT,GAAsB,GAAL,MAAK,CAAC,QAAD,I  
AAL,C;QACxC,OAAW,CAAC,WAAa,EAAd,MAAqB,CAAhC,GjDpIwC,MAAa,KAAK,UiDoIIB,MjDpIkB,EiDo  
IV,OjDpIU,CiDoI1D,GjDpIwC,MAAa,KAAK,UiDoIS,OjDpIT,EiDoIkB,MjDpIIB,C;;QiDsInD,Q;QAAA,IAAI,CA  
AC,WAAa,EAAd,MAAqB,CAAZB,C;UAAA,OAA4B,S;;uBjDpIiB,uB;UiDoIP,ajDrIM,sB;UiDqI5C,OjDtlIC,MA  
Aa,KAAK,kBAAC,MAAd,C;;QiDsI1D,W;;K;kFAKR,yB;MAAA,4C;MAAA,sC;QAaiE,6BAAW,CAAC,QAAD,I  
AAX,C;O;KAbjE,C;qECzKA,kC;MAII,OAAO,SAA8B,MAAK,WAAAL,C;K;uEAGzC,8C;MAII,OAAO,SAA8B,M  
AAK,WAAAL,EAakB,UAAIB,C;K;ICpCzC,iC;MACI,gBAAH,IAAI,OAAO,EAAG,GAAE,IAAI,IAAI,CAAC,CA  
AD,EAAL,EAAJ,CAAd,GAAyB,CAAhC,C;K;;IAKJ,sC;MACI,cAAO,QAAP,GAakB,QAAQ,Q;K;ICP9B,yC;K;;I  
AWA,+B;K;;4GAYA,yB;MAAA,gC;MAAA,yD;MAAA,sC;QAQI,OAAK,qBAAL,SAAK,iB;O;KART,C;ICPI,2B;  
MAAS,Q;MAAD,OAAwB,CAAvB,iEAAuB,Q;K;IAMhC,+B;MAAQ,iBAAU,SAAV,C;K;;;;;ICtB+B,4B;MACv  
C,8B;K;gEAAA,Y;MAAA,4B;K;2FAII,Y;MxGO4B,MAAM,yB;K;kCwGLiC,iB;MACI,OAAO,oCAA0B,oBAAU,  
KAAM,OAAhB,C;K;oCAGrC,Y;MAC+B,gB;MAAA,8FAA0B,C;K;oCAEzD,Y;MAEI,OAAO,oBAAQ,eAAR,C;



EG,eAAhB,UAAqC,M;QAHIC,SAIH,M,;MAJJ,a;K;IA7D+E,8C;MAAE,6B;K;IAGO,iD;MAAE,0B;K;IAME,kD;  
MAAE,8B;K;IAGZ,+C;MAAE,6B;K;IAGC,gD;MAAE,6B;K;IAGR,8C;MAAE,6B;K;IAGI,gD;MAAE,6B;K;IAG  
C,iD;MAAE,6B;K;IAGH,gD;MAAE,yB;K;IAGD,iD;MAAE,6B;K;IAGM,oD;MAAE,mC;K;IAGO,uD;MAAE,gC;  
K;IAGL,oD;MAAE,6B;K;IAGJ,oD;MAAE,6B;K;IAGE,qD;MAAE,8B;K;IAGR,mD;MAAE,4B;K;IAGJ,oD;MAA  
E,6B;K;IAGQ,qD;MAAE,8B;K;IAGC,sD;MAAE,+B;K,;IA5DvH,wC;MAAA,uC;QAAA,sB;OAAA,gC;K,;ICCA,  
2B;MAEW,Q;MAAA,IAAI,KAAY,SAAQ,MAAR,CAAhB,C;QACH,kBAAW,MAAX,C,;QAEA,kBAAW,MAAX  
,C,;MAHJ,W;K;IAOJ,8B;MAC4E,QAAM,QAAS,OAAf,C;aACx,E,C;UADwE,OACnE,WAAW,SAAS,CAAT,CA  
AX,C;aACL,C;UAFwE,OAEnE,+B;gBAFmE,OAGhE,iB,;K;IAGZ,oC;MAEU,IAAN,I;MAAA,QvEhB0C,OuEgB3  
B,CAAf,C;aACI,Q;UAA6B,OAAjB,8BAAiB,Y;UAA7B,K;aACA,Q;UAAy,OAAI,CAAY,C9DbhC,G8DamC,CA  
Af,MAAkC,CAAtC,GAAyC,8BAAiB,SAAID,GAAwE,8BAAiB,Y;UAArG,K;aACA,S;UAA8B,OAAjB,8BAAiB,  
a;UAA9B,K;aACA,U;UAA+B,OAAjB,8BAAiB,eAAgB,CAAY,OAA5B,C;UAA/B,K;gBAGQ,6B;YAAAsC,OAAj  
B,8BAAiB,kB;eACtC,0B;YAAmC,OAAjB,8BAAiB,e;eACnC,0B;YAAmC,OAAjB,8BAAiB,e;eACnC,2B;YAAo  
C,OAAjB,8BAAiB,gB;eACpC,yB;YAAkC,OAAjB,8BAAiB,c;eACiC,0B;YAAmC,OAAjB,8BAAiB,e;eACnC,2B;  
YAAoC,OAAjB,8BAAiB,gB;eACpC,4B;YAAqC,OAAjB,8BAAiB,iB;eACrC,6B,;eACA,sB;YAAkC,OAAjB,8BA  
AiB,W,;YAE9B,kBAAkB,MAAA,gBA Ae,CAAf,CAAkB,Y;YAE7C,oBAAgB,MAAhB,C;cAAiD,OAAjB,8BAAiB  
,S;iBACjD,oBAAgB,KAAhB,C;cAAgD,OAAjB,8BAAiB,e,;cAE5C,cAA0B,W;cAC1B,kBAAW,OAAAX,C,;UAX  
BxB,K,;MAAA,W;K;IAGCJ,4B;MAMW,Q;MAJP,IAAI,WAAW,MAAf,C;QAA6B,OAAO,8BAAiB,Y;OAErD,eA  
AsB,MAAY,W;MAE3B,IAAI,gBAAJ,C;QACH,IAAI,QAAS,SAAT,QA AJ,C;UACI,aAAa,qBAAiB,MAAJB,C;UA  
Cb,oBAAsB,M;UACtB,a,;UAES,OAAT,QAAS,S,;QAGb,4BAAiB,MAAJB,C,;MATJ,W;K;ICrCJ,0B;MAIL,sBAA  
Y,C;K;qEAcH,4B;MAIkE,iBAAY,KAAZ,C;K;2EAEIE,qB;MAI8D,gB;K;ICIDb,2C;MAC7C,qBAAwC,Q;K;iDA  
ExC,Y;MACmB,Q;MAAA,yB;MAAA,iB;QAAe,MAAM,6BAAsB,0CAAtB,C;OAApC,eAAe,I;MACf,qBAAc,I;M  
ACd,OAAO,QAAS,W;K,;ICLa,kD;MADrC,e;MACsC,0B;MAAyB,gB;MAD/D,iB;MAAA,uB;K;IAAA,mC;MA  
AA,sC;O;MAEI,qEAGW,CAHX,EAGc,IAHd,C;MAKA,iFAGiB,CAHjB,EAGoB,IAHpB,C;MAKA,iFAGiB,CAHj  
B,EAGoB,IAHpB,C;MAKA,iFAGiB,CAHjB,EAGoB,IAHpB,C;MAKA,+EAGgB,CAHhB,EAGmB,IAHnB,C;MA  
KA,yEAGa,CAHb,EAGgB,IAHhB,C;MAKA,iFAGiB,CAHjB,EAGoB,IAHpB,C;MAKA,6EAGe,CAHf,EAGkB,I  
AHIB,C;MAKA,6FAGuB,CAHvB,EAG0B,IAH1B,C;MAKA,yFAGqB,CAHrB,EAGwB,IAHxB,C;MAKA,4EAGc  
,EAHd,EAGkB,IAHIB,C;MAKA,0EAGa,EAHb,EAGiB,IAHjB,C;MAKA,gFAGgB,EAHhB,EAGoB,IAHpB,C;M  
AKA,8EAGe,EAHf,EAGmB,IAHnB,C;MAKA,wFAGoB,EAHpB,EAGwB,IAHxB,C;MAKA,gEAGQ,EAHR,EA  
GY,IAHZ,C;MAKA,8DAGO,EAHP,EAGW,IAHX,C;MAKA,wEAGY,EAHZ,EAGgB,IAHhB,C;MAKA,oEAGU,  
EAHV,EAGc,IAHd,C;MAKA,kFAGiB,EAHjB,EAGqB,IAHrB,C;MAKA,oFAGkB,EAHIB,EAGsB,IAHtB,C;MA  
KA,gFAGgB,EAHhB,EAGoB,IAHpB,C;MAKA,4FAGsB,EAHtB,EAG0B,IAH1B,C;MAKA,oFAGkB,EAHIB,EA  
GsB,IAHtB,C;MAKA,wEAGY,EAHZ,EAGgB,IAHhB,C;MAKA,gFAGgB,EAHhB,EAGoB,IAHpB,C;MAKA,gF  
AGgB,EAHhB,EAGoB,IAHpB,C;MAKA,0EAGa,EAHb,EAGiB,IAHjB,C;MAKA,oGAG0B,EAH1B,EAG8B,IAH  
9B,C;MAKA,gGAGwB,EAHxB,EAG4B,IAH5B,C;MAUA,oC;K;IA3JA,+C;MAAA,yB;MAAA,uC;K,;IAKA,qD;  
MAAA,yB;MAAA,6C;K,;IAKA,qD;MAAA,yB;MAAA,6C;K,;IAKA,qD;MAAA,yB;MAAA,6C;K,;IAKA,oD;MA  
AA,yB;MAAA,4C;K,;IAKA,iD;MAAA,yB;MAAA,yC;K,;IAKA,qD;MAAA,yB;MAAA,6C;K,;IAKA,mD;MAAA  
,yB;MAAA,2C;K,;IAKA,2D;MAAA,yB;MAAA,mD;K,;IAKA,yD;MAAA,yB;MAAA,iD;K,;IAKA,kD;MAAA,yB  
;MAAA,0C;K,;IAKA,iD;MAAA,yB;MAAA,yC;K,;IAKA,oD;MAAA,yB;MAAA,4C;K,;IAKA,mD;MAAA,yB;M  
AAA,2C;K,;IAKA,wD;MAAA,yB;MAAA,gD;K,;IAKA,4C;MAAA,yB;MAAA,oC;K,;IAKA,2C;MAAA,yB;MAA  
A,mC;K,;IAKA,gD;MAAA,yB;MAAA,wC;K,;IAKA,8C;MAAA,yB;MAAA,sC;K,;IAKA,qD;MAAA,yB;MAAA,  
6C;K,;IAKA,sD;MAAA,yB;MAAA,8C;K,;IAKA,oD;MAAA,yB;MAAA,4C;K,;IAKA,0D;MAAA,yB;MAAA,kD;  
K,;IAKA,sD;MAAA,yB;MAAA,8C;K,;IAKA,gD;MAAA,yB;MAAA,wC;K,;IAKA,oD;MAAA,yB;MAAA,4C;K,;  
IAKA,oD;MAAA,yB;MAAA,4C;K,;IAKA,iD;MAAA,yB;MAAA,yC;K,;IAKA,8D;MAAA,yB;MAAA,sD;K,;IAK  
A,4D;MAAA,yB;MAAA,oD;K;8CAKA,gB;MAG2D,OAAK,iBAAL,IAAK,CAAL,KAA2B,IAAK,c;K;IAE3F,kC;  
MAAA,sC;K;uDACI,oB;MAEQ,IADE,QACF,IAAG,CAAH,IADE,QACF,IAAM,EAAN,C;QADJ,OACgB,sBAAS  
,QAAT,C;WACZ,IAFE,QAEF,IAAG,EA AH,IAFE,QA EF,IAAO,EAAP,C;QAFJ,OAEiB,sBAAS,WAAW,CAAX,I  
AAT,C,;QACL,MAAM,gCAAyB,eAAY,QA AZ,qBAAzB,C;K,;IAL1B,8C;MAAA,yB;MAAA,6C;QAAA,4B;OA  
AA,sC;K,;IA7JJ,+B;MAAA,+yC;K,;IAAA,oC;MAAA,a;AAAA,Y;UAAA,4C;aAAA,kB;UAAA,kD;aAAA,kB;UA

AA,kD;aAAA,kB;UAAA,kD;aAAA,iB;UAAA,iD;aAAA,c;UAAA,8C;aAAA,kB;UAAA,kD;aAAA,gB;UAAA,gD;  
aAAA,wB;UAAA,wD;aAAA,sB;UAAA,sD;aAAA,e;UAAA,+C;aAAA,c;UAAA,8C;aAAA,iB;UAAA,iD;aAAA,g  
B;UAAA,gD;aAAA,qB;UAAA,qD;aAAA,S;UAAA,yC;aAAA,Q;UAAA,wC;aAAA,a;UAAA,6C;aAAA,W;UAAA  
,2C;aAAA,kB;UAAA,kD;aAAA,mB;UAAA,mD;aAAA,iB;UAAA,iD;aAAA,uB;UAAA,uD;aAAA,mB;UAAA,m  
D;aAAA,a;UAAA,6C;aAAA,iB;UAAA,iD;aAAA,iB;UAAA,iD;aAAA,c;UAAA,8C;aAAA,2B;UAAA,2D;aAAA,y  
B;UAAA,yD;gBAAA,6D;;K;;ICKiD,2C;uBAA+B,O;;K;;IAC5E,8C;MAAA,ke;MAAuB,qCAAK,IAAL,C;MAAv  
B,Y;K;ICD8B,gC;MAe9B,gBAAiC,YAAY,SAAhB,GAA2B,OAA3B,GAAwC,E;K;uFAGjE,Y;MAAQ,OAAO,aA  
AY,O;K;yCAE/B,iB;MACW,gBAAP,a;MpGoGG,Q;MAAA,IoGpGc,KpGoGV,IAAS,CAAT,IoGpGU,KpGoGI,IA  
AS,2BAA3B,C;QAAA,OAAc,qBoGpGxB,KpGoGwB,C;;QoGpGf,MAAM,8BAA0B,mCAAyB,WAAzB,MAA1  
B,C;;MAAhC,W;K;kDAEJ,gC;MAAgF,OAAA,azGiMY,WyGjMK,UzGiML,EyGjMiB,QzGiMjB,C;K;6CyG/L5F,i  
B;MACI,qCAAU,KAAV,C;MACA,OAAO,I;K;6CAGX,iB;MACI,iBAAGB,SAAN,KAAM,C;MACHB,OAAO,I;K;  
6CAGX,uC;MACI,OAAA,IAAK,qBAAY,wBAAS,MAArB,EAA6B,UAA7B,EAAyC,QAAzC,C;K;sCAET,Y;MA  
ayB,UAEK,M;MAL1B,eAAe,E;MACf,YAAY,aAAO,OAAP,GAAgB,CAAhB,I;MACZ,OAAO,SAAS,CAAhB,C;  
QACI,UAAU,0BAAO,YAAP,EAAO,oBAAP,Q;QACV,IAAQ,eAAJ,GAAl,CAAJ,IAAwB,SAAS,CAArC,C;UACI  
,WAAW,0BAAO,cAAP,EAAO,sBAAP,U;UACX,IAAS,gBAAL,IAAK,CAAT,C;YACI,WAAW,+BAAW,iBAAX,  
wBAAkB,gBAAIB,C;;YAEX,WAAW,+BAAW,gBAAX,wBAAiB,iBAAjB,C;;;UAGf,gCAAY,GAAZ,C;;MAGR,  
gBAAS,Q;MACT,OAAO,I;K;6CAGX,iB;MAOI,iBAAGB,SAAN,KAAM,C;MACHB,OAAO,I;K;6CAGX,iB;MAQ  
I,iBAAU,K;MACV,OAAO,I;K;6CAGX,iB;MAQI,iBAAGB,eAAN,KAAM,C;MACHB,OAAO,I;K;6CAGX,iB;MA  
C2C,2BAAO,KAAP,C;K;6CAE3C,iB;MAOI,gBAAA,IAAK,SAAL,IAAe,wBAAS,MAAxB,C;MACA,OAAO,I;K;  
uCAGX,Y;MAU6B,kB;K;qDAE7B,2B;K;8CAcA,kB;MAO0C,OAAA,IAAY,SAAY,SAAQ,MAAR,C;K;8CAEIE,  
8B;MAQ2D,OAAA,IAAY,SAAY,SAAQ,MAAR,EAAgB,UAAhB,C;K;kDAEnF,kB;MAQ8C,OAAA,IAAY,SAA  
Y,aAAY,MAAZ,C;K;kDAEIE,8B;MASI,IAAI,MnGuGwC,YAAU,CmGvGID,IAAoB,aAAa,CAArC,C;QAAwC,O  
AAO,E;MAC/C,OAAO,IAAY,SAAY,aAAY,MAAZ,EAAoB,UAApB,C;K;4CAGnC,wB;MAWI,oCAAA,4BAAm  
B,KAAAnB,EAA0B,WAA1B,C;MAEb,gBAAS,azGmB+E,WyGnB9D,CzGmB8D,EyGnB3D,KzGmB2D,CyGnB/E,  
YAA6B,KAA7B,IAAqC,azGgB2B,WyGhBV,KzGgBU,C;MyGfzE,OAAO,I;K;6CAGX,wB;MAQI,oCAAA,4BA  
mB,KAAAnB,EAA0B,WAA1B,C;MAEb,gBAAS,azGK+E,WyGL9D,CzGK8D,EyGL3D,KzGK2D,CyGL/E,uBAA6  
B,kBAA7B,IAAqC,azGE2B,WyGFV,KzGEU,C;MyGDzE,OAAO,I;K;6CAGX,wB;MAUI,oCAAA,4BAAmB,KAA  
nB,EAA0B,WAA1B,C;MAEb,gBAAS,azGX+E,WyGW9D,CzGX8D,EyGW3D,KzGX2D,CyGW/E,GAAMC,eAA  
N,KAAM,CAAnC,GAAsD,azGdU,WyGcO,KzGdP,C;MyGezE,OAAO,I;K;6CAGX,wB;MAAI,oCAAA,4BAAmB,  
KAAAnB,EAA0B,WAA1B,C;MAEb,gBAAS,azG9B+E,WyG8B9D,CzG9B8D,EyG8B3D,KzG9B2D,CyG8B/E,GA  
AmC,SAAN,KAAM,CAAnC,GAAgD,azGjCgB,WyGiCC,KzGjCD,C;MyGkCzE,OAAO,I;K;6CAGX,wB;MAWI,  
oCAAA,4BAAmB,KAAAnB,EAA0B,WAA1B,C;MAEb,gBAAS,azG/C+E,WyG+C9D,CzG/C8D,EyG+C3D,KzG/C2  
D,CyG+C/E,GAAMC,SAAN,KAAM,CAAnC,GAAgD,azGIdgB,WyGkDC,KzGIDD,C;MyGmDzE,OAAO,I;K;6C  
AGX,wB;MACuD,2BAAO,KAAP,EAAC,KAAd,C;K;6CAEvD,wB;MAUI,oCAAA,4BAAmB,KAAAnB,EAA0B,W  
AA1B,C;MAEb,eAAe,wBAAS,M;MACxB,gBAAC,IAAK,SzGnEqE,WyGmEpD,CzGnEoD,EyGmEjD,KzGnEiD,  
CyGmEIE,GAAC,QAAIC,GAA6C,IAAK,SzGtES,WyGsEQ,KzGtER,C;MyGuEzE,OAAO,I;K;gDAGX,qB;MAC  
I,IAAI,YAAY,CAAhB,C;QACI,MAAM,gCAAyB,0BAAuB,SAAvB,MAAzB,C;OAGV,IAAI,aAAa,WAAjB,C;Q  
ACI,gBAAS,azG1F2E,WyG0F1D,CzG1F0D,EyG0FvD,SzG1FuD,C;;QyG4FpF,aAAU,WAAV,MAAuB,SAAvB,  
M;UACI,qCAAU,CAAV,C;;;K;gDAKZ,sB;MAQI,oCAAA,4BAAmB,UAAAnB,EAA+B,WAA/B,C;MAEb,OAAO,a  
zG/GkE,WyG+GjD,UzG/GiD,C;K;gDyGkH7E,gC;MAQI,oCAAA,4BAAmB,UAAAnB,EAA+B,QAA/B,EAAyC,W  
AAzC,C;MAEb,OAAO,azGzHiF,WyGyHhE,UzGzHgE,EyGyHpD,QzGzHoD,C;K;yCyG4H5F,Y;K;uCAcA,Y;MA  
AkC,oB;K;oCAEIC,Y;MAOI,gBAAS,E;MACT,OAAO,I;K;0CAGX,wB;MAQI,oCAAA,2BAAkB,KAAIB,EAAyB,  
WAAzB,C;MAEb,gBAAS,azGjK+E,WyGiK9D,CzGjK8D,EyGiK3D,KzGjK2D,CyGiK/E,uBAA6B,kBAA7B,IAA  
qC,azGpK2B,WyGoKV,QAAQ,CAAR,IzGpKU,C;K;+CyGuK7E,uC;MAYI,yBAAkB,UAAIB,EAA8B,QAA9B,E  
AAwC,WAAxC,C;MAEA,gBAAC,IAAK,SzGILqE,WyGkLpD,CzGILoD,EyGkLjD,UzGILiD,CyGkL1E,GAAC,  
KAAvC,GAA+C,IAAK,SzGrLO,WyGqLU,QzGrLV,C;MyGsLzE,OAAO,I;K;kDAGX,wC;MACI,IAAI,aAAa,CA  
Ab,IAAkB,aAAa,MAAnC,C;QACI,MAAM,8BAA0B,iBAAC,UAAAd,kBAAMC,MAA7D,C;OAEV,IAAI,aAAa,QA  
AjB,C;QACI,MAAM,gCAAyB,gBAAa,UAAAb,qBAAqC,QAArC,MAAzB,C;Q;+CAId,iB;MAYI,oCAAA,2BAAkB

,KAAIB,EAAyB,WAAzB,C;MAEb,gBAAS,azG7M+E,WyG6M9D,CzG7M8D,EyG6M3D,KzG7M2D,CyG6M/E,GAA6B,azGhNmC,WyGgNIB,QAAQ,CAAR,IzGhNkB,C;MyGiNzE,OAAO,I;K;kDAGX,gC;MAWI,yBAaKB,UAAIB,EAA8B,QAA9B,EAAwC,WAAxC,C;MAEA,gBAAS,azG9N+E,WyG8N9D,CzG9N8D,EyG8N3D,UzG9N2D,CyG8N/E,GAAkC,azGjO8B,WyGiOb,QzGjOa,C;MyGkOzE,OAAO,I;K;kDAGX,gE;MAC+c,iC;QAAA,oBAAyB,C;MAAG,0B;QAAA,aAAkB,C;MAAG,wB;QAAA,WAAgB,IAAK,O;MAKIF,IACf,I;MALhB,oCAAa,4BAAmB,UAAAnB,EAA+B,QAA/B,EAAyC,WAAzC,C;MACb,oCAAa,4BAAmB,iBAAnB,EAAc,oBAAoB,QAApB,GA A+B,UAA/B,IAAtC,EAAiF,WAAy,OAA7F,C;MAEb,eAAe,iB;MACf,iBAAc,UAAAd,UAA+B,QAA/B,U;QACI,YAAy,eAAZ,EAAy,uBAAZ,UAA0B,yBAAO,KAAP,C;;K;kDAIIC,uC;MAcI,iBAAGB,iBAAN,KAAM,EAAe,UAAf,EAA2B,QAA3B,C;MACHB,OAAO,I;K;kDAGX,uC;MAYI,gBAAgB,KAAM,W;MACtB,oCAAa,4BAAmB,UAAAnB,EAA+B,QAA/B,EAAyC,SAAU,OAAAnD,C;MAEb,iBAAU,SzG3R8E,WyG2R1D,UzG3R0D,EyG2R9C,QzG3R8C,C;MyG4RxF,OAAO,I;K;kDAGX,8C;MAgBI,oCAAa,4BAAmB,KAAnB,EAA0B,IAAK,OAA/B,C;MAEb,gBAAS,azGjT+E,WyGiT9D,CzGjT8D,EyGiT3D,KzGjT2D,CyGiT/E,GAAmC,iBAAN,KAAM,EAAe,UAAf,EAA2B,QAA3B,CAAnC,GAA0E,azGpTV,WyGoT2B,KzGpT3B,C;MyGqTzE,OAAO,I;K;kDAGX,8C;MAgBI,oCAAa,4BAAmB,KAAnB,EAA0B,WAA1B,C;MAEb,gBAAgB,KAAM,W;MACtB,oCAAa,4BAAmB,UAAAnB,EAA+B,QAA/B,EAAyC,SAAU,OAAAnD,C;MAEb,gBAAS,azG1U+E,WyG0U9D,CzG1U8D,EyG0U3D,KzG1U2D,CyG0U/E,GAA6B,SzG1UkD,WyG0U9B,UzG1U8B,EyG0UIB,QzG1UkB,CyG0U/E,GAAyE,azG7UT,WyG6U0B,KzG7U1B,C;MyG8UzE,OAAO,I;K;;IALiBX,6C;MAAA,uD;MAKoC,2B;MALpC,Y;K;IAQA,8C;MAAA,uD;MAC4C,0BAAK,OAAQ,WAAb,C;MAD5C,Y;K;IAGA,qC;MAAA,uD;MACuB,0BAAK,EAAL,C;MADvB,Y;K;2EA4hBJ,qB;MAOgE,OAAA,SAAK,Q;K;uEAeRE,mC;MAQ+E,SAAK,aAAI,KAAJ,EAAW,KAAX,C;K;+EAepF,kD;MAaI,OAAA,SAAK,kBAAS,UAAAT,EAAqB,QAArB,EAA+B,KAA/B,C;K;+EAET,4B;MAY6E,OAAA,SAAK,kBAAS,KAAAT,C;K;qFAEIF,2C;MAWoG,OAAA,SAAK,qBAAY,UAAZ,EAAwB,QAAxB,C;K;uFAEzG,2E;MAe2E,iC;QAAA,oBAAyB,C;MAAG,0B;QAAA,aAAkB,C;MAAG,wB;QAAA,WAAgB,SAAK,O;MAC7I,SAAK,qBAAY,WAAZ,EAAyB,iBAAzB,EAA4C,UAA5C,EAAwD,QAAxD,C;K;qFAET,kD;MAeI,OAAA,SAAK,qBAAY,KAAZ,EAAMb,UAAAnB,EAA+B,QAA/B,C;K;uFAET,kD;MAaI,OAAA,SAAK,qBAAY,KAAZ,EAAMb,UAAAnB,EAA+B,QAA/B,C;K;qFAET,yD;MAiBI,OAAA,SAAK,qBAAY,KAAZ,EAAMb,KAAnB,EAA0B,UAA1B,EAAc,QAAtC,C;K;uFAET,yD;MAiBI,OAAA,SAAK,qBAAY,KAAZ,EAAMb,KAAnB,EAA0B,UAA1B,EAAc,QAAtC,C;K;qF1GhsBT,qB;MAMoD,OA6BW,8BAAY,cAfrB,YAAy,CAAZ,C;K;yFAZtD,qB;MAYsD,OAeS,8BAAY,cAfrB,YAAy,CAAZ,C;K;iFAEtD,qB;MAaoD,OAAW,8BAAY,c;K;qFAE3E,yB;MAAA,uD;MAAA,4B;QAMoD,+B;O;KANpD,C;IAQA,kC;MAYI,gBAiB2D,8BAAY,c;MAhBvE,OAAW,SAAU,OAAV,GAAmB,CAAvB,GAA0B,SAAI1B,GAAoC,qBAAU,CAAV,C;K;iFAG/C,qB;MAaoD,OAAW,8BAAY,c;K;IAE3E,kC;MAU+C,mC;K;IAE/C,oC;MAGoD,QAAQ,cAAA,sCAAK,mBAAL,EAAyB,sCAAK,mBAA9B,CAAR,6B;K;IAEpD,mC;MAGmD,QAAQ,cAAA,sCAAK,kBAAL,EAAwB,sCAAK,kBAA7B,CAAR,6B;K;IAO/C,iC;MAAQ,OAAA,oCAAa,iBAAQ,2BAAR,C;K;IAEzB,8B;MAOI,IAAI,YAAO,GAAX,C;QACI,OAAO,I;OAEX,OAAO,gCAA8C,mD;K;IAGzD,6B;MAUI,IAAI,CAAQ,kBAAK,GAAL,CAAR,iCAAoB,CAAQ,kBAAK,EAAL,CAAR,6BAAxB,C;QACI,OAAO,I;OAEX,IAAI,YAAO,GAAX,C;QACI,OAAO,K;OAEX,OAAO,uB;K;IAGX,oC;MAUI,IAAI,CAAQ,kBAAK,GAAL,CAAR,iCAAoB,CAAQ,kBAAK,EAAL,CAAR,6BAApB,IAAwC,CAAQ,kBAAK,EAAL,CAAR,6BAA5C,C;QACI,OAAO,I;OAEX,IAAI,YAAO,GAAX,C;QACI,OAAO,K;OAGX,OAAO,0BAAiB,uB;K;IAG5B,4B;MASI,IAAI,CAAQ,kBAAK,EAAL,CAAR,6BAAJ,C;QACI,OAAO,I;OAEX,IAAI,YAAO,GAAX,C;QACI,OAAO,K;OAEX,OAAO,sB;K;IAGX,gC;MAUI,IAAI,CAAQ,kBAAK,EAAL,CAAR,6BAAJ,C;QACI,OAAO,I;OAEX,IAAI,YAAO,GAAX,C;QACI,OAAO,K;OAEX,OAAO,0B;K;IAGX,gC;MASI,IAAI,YAAO,GAAX,C;QACI,OAAO,K;OAEX,OAAO,gCAAoD,yD;K;IAG/D,iC;MAUI,OAAO,aAAQ,EAAR,IAAoB,CAAQ,mBAAU,GAAV,CAAR,6B;K;IAG/B,iC;MAMiD,kC;K;iF2GtPjD,yB;MAAA,+C;MAAA,4B;QAMuD,OAkK,UAAAL,SAAK,C;O;KAN5D,C;IAQA,gC;MAMiD,4B;MAAA,S;QAAgB,cAAA,S1G4LC,c0G5LD,EAAoB,MAApB,C;OAAhB,W;K;IAEjD,6B;MAI0C,Q;MAAA,yDAaKB,kBAaKB,SAAI1B,C;K;IAE5D,oC;MAKoD,Q;MAAA,yCAAa,KAAb,oBAAuB,kBAaKB,SAAI1B,C;K;IAG3E,8B;MAI4C,Q;MAAA,0DAAMb,kBAaKB,SAAI1B,C;K;IAE/D,qC;MAKsD,Q;MAAA,0CAAc,KAAAd,oBAAwB,kBAaKB,SAAI1B,C;K;IAE9E,0B;MAIwC,Q;MAAA,wDAAiB,kBAaKB,SAAI1B,C;K;IAEzD,mC;MAKkD,Q;MAAA,wCAAY,KAAZ,oBAAsB,kBAaKB,SAAI1B,C;K;IAExE,2B;MAI0C,Q;MAAA,y

DAAkB,kBAaKB,SAaIB,C;K;IAE5D,oC;MAKoD,Q;MAAA,yCAAA,KAAb,oBAaUB,kBAaKB,SAaIB,C;K;IAE3E,6B;MAIyF,kBAa1C,CAAO,S;MACID,IAAO,QpHeD,WoHfC,CAAH,IAAc,CAAM,kBAaPB,KpHeE,WoHf6B,KAAM,GAAN,IAaKB,kBAaJD,CAAJ,C;QACI,4B;MAFsC,OpHiBnC,W;K;6EoHZX,yB;MAAA,6C;MAAA,4B;QAKmD,0B;O;KALnD,C;IAOA,mC;MAIgG,kBAa1C,CAAO,S;MAAR,OACjD,EAAK,QpH2BgB,WoH3BhB,CAAH,IAAc,CAAM,kBAaPB,KpH2BmB,WoH3BY,KAAM,GAAN,IAaKB,kBAaJD,CAAF,CpH2BO,GAAqB,WAArB,GAA+B,I;K;yFoHxB1C,yB;MAAA,yD;MAAA,4B;QAK0D,gC;O;KAL1D,C;IFAOA,yB;MAAA,6C;MAAA,mC;QAO6D,OAAa,SAAR,SAAQ,EAAS,KAAT,C;O;KAP1E,C;IFASA,yB;MAAA,6C;MAAA,mC;QAO8D,OAAa,SAAR,SAAQ,EAAS,KAAT,C;O;KAP3E,C;IASA,sC;MAMqD,OAAA,SAAY,UAAS,WAAW,KAAX,CAAT,C;K;IAEjE,4B;MAAsC,QAAM,S1G4EsB,c0G5E5B,C;aACIC,K;aAAA,M;aAAA,M;UADkC,OACT,I;gBADS,OAE1B,K;;K;IAGZ,2B;MAKI,IAAI,EAAU,CAAV,sBAaA,EAAb,CAAJ,C;QACI,MAAM,gCAAYB,WAAQ,KAAR,kCAAzB,C;OAEV,OAAO,K;K;IAGX,8B;MAA2D,Q;MACvD,YAAQ,EAAR,IAAe,QAAQ,EAaVb,C;QAA8B,cAAO,E;WACrC,YAAQ,EAAR,IAAe,QAAQ,EAaVb,C;QAA8B,cAAO,EAAP,GAAa,EAAb,I;WAC9B,YAAQ,EAAR,IAAe,QAAQ,EAaVb,C;QAA8B,cAAO,EAAP,GAAa,EAAb,I;WAC9B,WAAO,GAAP,C;QAAMb,S;WACnB,YAAQ,KAAR,IAAoB,QAAQ,KAA5B,C;QAAwC,cAAO,KAAP,GAaKB,EAaIB,I;WACxC,YAAQ,KAAR,IAAoB,QAAQ,KAA5B,C;QAAwC,cAAO,KAAP,GAaKB,EAaIB,I;;QAC3B,sBAAL,IAAK,C;Mph9CN,a;MoHuCgD,OAQ/C,WAAJ,GAAiB,EAajB,GAAyB,E;K;ICIjG,2C;MAHpC,e;MAGqC,kB;MAHrC,iB;MAAA,uB;K;IAAA,kC;MAAA,qC;O;MAII,qEACY,GADZ,C;MAEA,iEAIU,GAJV,C;K;;IAFA,+C;MAAA,wB;MAAA,uC;K;;IAEA,6C;MAAA,wB;MAAA,qC;K;;IANJ,8B;MAAA,mF;K;;IAAA,mC;MAAA,a;aAAA,a;UAAA,4C;aAAA,W;UAAA,0C;gBAAA,4D;;K;;IAawG,4B;MAAE,OAAA,EAAG,M;K;IAA7G,qC;MAAQE,iCAAA,EAAb,EAa0B,OAA1B,0BAAmC,cAAnC,C;K;IAQIC,2B;MAAC,kB;K;;sCALpC,Y;MAKoC,iB;K;wCALpC,iB;MAAA,sBAKoC,qCALpC,C;K;oCAAA,Y;MAAA,OAKoC,iDALpC,M;K;oCAAA,Y;MAAA,c;MAKoC,sD;MALpC,a;K;kCAAA,iB;MAAA,2IAKoC,sCALpC,G;K;IAqB0B,iC;MA8PtB,6B;MArPA,eACoC,O;MACpC,eACsD,QAAR,OAAQ,C;MACtD,uBAAoC,WAAO,OAAP,EAawB,QAAR,OAAQ,EAQ,IAAR,CAAxB,C;MACpC,6BAA2C,I;MAI3C,oCAAKD,I;K;0CAHID,Y;MACI,Q;MAAA,U;MAAA,gD;QAAA,a;;QAA8D,gBAaVc,WAAO,YAAP,EAawB,QAAR,YAAQ,EAQ,IAAR,CAAxB,C;QAA8C,6BrHmCnE,S;QqHnCF,SrHoCG,S;;MqHpCH,a;K;iDAGJ,Y;MACI,Q;MAAA,U;MAAA,uD;QAAA,a;;QrHVG,gB;QqHWC,IAAY,aAAR,YAAQ,EAawB,EAAX,CAAR,IAAmC,WAAW,YAAQ,EAAS,EAAT,CAAvC,C;UAAA,eACI,oB;;UAEA,OAAO,WAAO,MAA2B,UAAf,YAAR,YAAQ,qBAAU,EAaV,EAaE,qBAAQ,EAAR,EAa3B,MAAP,EAa2D,QAAR,YAAQ,EAQ,IAAR,CAA3D,C;QACb,4B;QAAO,oCrH0BP,S;QqH/BF,SrHgCG,S;;MqHhCH,a;K;sCAQJ,iB;MAEkB,MAAd,oBAAc,C;MACd,YAAY,oBAAc,MAAK,KAAM,WAAAX,C;MAC1B,OAAO,iBAaIB,KAAM,MAAN,KAAe,CAAhC,IAAqC,oBAAc,UAAAd,KAA2B,KAAM,O;K;8CAGjF,iB;MAEkB,MAAd,oBAAc,C;MACd,OAAO,oBAAc,MAAK,KAAM,WAAAX,C;K;wCAGzB,wB;MAGI,IAAIL,QAAQ,CAAR,IAAa,QAAQ,KAAM,OAA/B,C;QACI,MAAM,8BAA0B,0BAAuB,KAAvB,wBAA8C,KAAM,OA9E,C;OAEV,cAAc,0B;MACd,oBAAoB,K;MACpB,OAAO,OAAQ,MAAK,KAAM,WAAAX,C;K;mCAGnB,6B;MAS4C,0B;QAAA,aAaKB,C;MAC1D,IAAI,aAAa,CAAb,IAaKB,aAAa,KAAM,OAAzC,C;QACI,MAAM,8BAA0B,gCAA6B,UAA7B,wBAAYD,KAAM,OAAzF,C;OAEV,OAAqB,SAAd,oBAAc,EAAS,KAAM,WAAf,EAa2B,UAA3B,EAaUC,oBAAvC,C;K;IAeG,6E;MAAA,mB;QAAE,+BAAK,aAAL,EAAY,kBAAZ,C;O;K;IAA2B,uC;MAAW,OAAA,KAAM,O;K;sCAZ1E,6B;MAQ+C,0B;QAAA,aAaKB,C;MAC7D,IAAI,aAAa,CAAb,IAaKB,aAAa,KAAM,OAAzC,C;QACI,MAAM,8BAA0B,gCAA6B,UAA7B,wBAAYD,KAAM,OAAzF,C;OAEV,OAAO,mBAaIB,6CAAJB,EAa8C,sBAA9C,C;K;0CAGX,iB;MAMI,OAA2B,SAA3B,iCAA2B,EAAS,KAAM,WAAf,EAa2B,CAA3B,EAa8B,oBAA9B,C;K;sCAE/B,wB;MAGI,IAAI,QAAQ,CAAR,IAAa,QAAQ,KAAM,OAA/B,C;QACI,MAAM,8BAA0B,0BAAuB,KAAvB,wBAA8C,KAAM,OAA9E,C;OAEV,OAA2B,SAAPB,0BAAoB,EAAS,KAAM,WAAf,EAa2B,KAA3B,EAaKC,oBAaIC,C;K;IA4BL,mD;MAAA,qB;QAAE,2BAAoB,EAAPB,EAawB,mBAAXB,C;O;K;sCAxB5B,8B;MAqBI,IAAI,CAAA,YAAZ,WAAZ,EAAS,EAAT,CAAb,IAA+B,CAAA,YAAZ,WAAZ,EAAS,EAAT,CAAhD,C;QACI,OAAO,KAAM,W3G2E4E,S2G3EnD,oB3G2EmD,E2G3EpC,W3G2EoC,C;O2GzE7F,OAAO,qBAAQ,KAAR,EAaE,iCAAf,C;K;sCAGX,4B;MAMI,YAAY,kBAAK,KAAL,C;MACZ,IAAI,aAAJ,C;QAAMb,OAAO,KAAM,W;MAEHc,gBAAGB,C;MACHb,aAAa,KAAM,O;MACnB,SAAS,mBAAc,MAAd,C;;QAEI,iBAaIB,oB;QACjB,EAAG,gBAAO,KAAP,EAAC,SAAd,EAAY,UAAW,MAAM,MAA1C,C;QACH,EAAG,gBAAO,UAAU,UAAV,CAAP,C;QACH,YAAY,UAAW,MAAM,aAAjB,GAAGC,CAAhC,I;QACZ,QAAQ,UAAW,O;;MA

Cd,oBAAY,MAAZ,IAAsB,aAAtB,C;MAET,IAAI,YAAY,MAAhB,C;QACI,EAAG,gBAAO,KAAP,EAAC,SAAd,EAAYB,MAAZB,C;OAGP,OAAO,EAAG,W;K;2CAGd,8B;MA0BgB,Q;MALZ,IAAI,CAAa,YAAZ,WAAAY,EAAS,EAAT,CAAb,IAA+B,CAAa,YAAZ,WAAAY,EAAS,EAAT,CAAhD,C;QACI,uBAA+B,QAAR,YAAQ,EAAQ,GAR,C;QAC/B,OAAO,KAAM,W3GoB4E,S2GpBnD,WAAO,YAAP,EAAGB,gBAAhB,C3GoBmD,E2GpBhB,W3GoBgB,C;O2GjBjF,yBAAK,KAAL,C;MAAA,iB;QAAe,OAAO,KAAM,W;OAAxC,YAAY,I;MCoLO,gBAAhB,sB;MDjLC,yBrG2LgF,0BqG3LzD,CrG2LyD,EqG3LhD,WAAAM,MrG2L0C,CAAkC,WqG3LIH,C;MACA,yBAAO,uCAAP,C;MACA,yBrGyLgF,0BqGzLnD,WAAAM,KAAZ,GAAMB,CAAnB,IrGyLyD,EqGzL7B,YrGyL6B,CAAKC,WqGzLIH,C;MAHJ,OrHIJG,SsHoUqC,W;K;oCD3K5C,wB;MAO6C,qB;QAAA,QAAa,C;MAMxC,Q;MALd,wBAAwB,KAAxB,C;MrHrIG,SqHsIW,qBAAQ,KAAR,C;MAAd,cAAuC,UAAS,CAAb,GAAgB,EAAhB,GAA2B,OA AH,EAAG,EAAK,QAAQ,CAAR,IAAL,C;MAC9D,ahI3JgD,gB;MgI4JhD,gBAAgB,C;MAEF,yB;MAAd,OAAc,cAAd,C;QAAc,uB;QACV,MAAO,WAAU,mBAAN,KAAM,EAAY,SAAZ,EAAuB,KAAM,MAAM,MAAnC,CAA0C,WAApD,C;QACP,YAAY,KAAM,MAAM,aAAZ,GAA2B,CAA3B,I;;MAEhB,MAAO,WAAU,mBAAN,KAA M,EAAY,SAAZ,EAAuB,KAAM,OAA7B,CAAqC,WAA/C,C;MACP,OAAO,M;K;IAGBS,yI;MAAA,wC;MAAA,6 B;MAAA,yB;MAAA,0C;MAAA,oC;MAAA,0C;MAAA,yB;MAAA,6B;MAAA,8B;MAAA,8B;MAAA,kC;K;;;gE AAA,Y;;;;iCACA,mCAAK,wBAAL,C;cACZ,IAAI,4BAAiB,6BAAS,CAA9B,C;gBACI,gB;gCAAA,iCAAM,wB AAM,WAAZ,O;oBAAA,2C;yBAAA,yB;gBAAA,Q;;gBADJ,gB;,,,,;cAEI,M;;qCAGY,C;sCACC,C;cAEjB,gB;,,;sC ACqB,+B;cACjB,gB;8BAAA,iCrGuI4E,mBqGvItE,wBrGuIsE,EqGvItD,oBrGuIsD,EqGvI3C,qBAAW,MAAM,Mr GuI0B,CAAKC,WqGvI9G,O;kBAAA,2C;uBAAA,yB;cAAA,Q;;cACA,uBAAY,qBAAW,MAAM,aAAjB,GAAgC, CAAhC,I;cACZ,mBAAQ,qBAAW,O;cAJvB,KAKS,qDALT,EAKS,qBALT,OAKyB,2BAAQ,CAAR,IALzB,KAK sC,gBALtC,S;gBAAA,gB;,,;cAAA,gB;,,;cAOA,gB;8BAAA,iCrGkIgf,mBqGIIIE,wBrGkI0E,EqGIIID,oBrGkI0D,E qGII/C,wBAAM,OrGkIyC,CAAKC,WqGIIH,O;kBAAA,2C;uBAAA,yB;cAAA,Q;;cAhBA,OAgBA,a,,,,,,;K;I AjBY,sF;MAAA,yD;uBAAA,6H;YAAA,S;iBAAA,Q;;iBAAA,uB;O;K;8CAbpB,wB;MAUuD,qB;QAAA,QAAa,C ;MACH,e,wBAAwB,KAAxB,C;MAEA,OAAO,SAAS,gDAAT,C;K;+BAsBX,Y;MAMyC,OAAA,oBAAc,W;K;IAE vD,2B;MAAA,+B;MAmBI,uBAA4B,WAAO,uBAAP,EAAiC,GAAjC,C;MAC5B,2BAAgC,WAAO,SAAP,EAAo B,GAApB,C;MAGhC,iCAAsC,WAAO,KAAP,EAAb,GAAjB,C;K;oDAAtBtC,mB;MAIwD,oBAAM,oBAAO,OAA P,CAAN,C;K;+CAExD,mB;MAIoD,OAAA,O3GnEyC,S2GmEnB,oB3GnEmB,E2GmEJ,M3GnEI,C;K;0D2GqE7F ,mB;MAI+D,OAAA,O3GzE8B,S2GyER,wB3GzEQ,E2GyEW,M3GzEX,C;K;gE2G8E7F,mB;MAAgE,OAAA,O3 G9E6B,S2G8EP,8B3G9EO,E2G8EkB,M3G9EIB,C;K;;I2GwDjG,uC;MAAA,sC;QAAA,qB;OAAA,+B;K;;IA5PA, 4C;MAAA,+C;MACkE,kBAAK,OAAL,EAAC,MAAM,MAAN,CAAd,C;MADIE,Y;K;IAGA,sC;MAAA,+C;MAC 6C,kBAAK,OAAL,EAAC,UAAd,C;MAD7C,Y;K;IA4RO,kG;MAAA,kC;MAAA,8C;MAAA,kC;MAAA,kC;MAC H,uBAA+B,a;MAI/B,sF;MAOA,sBAA0C,I;K;+FAXI,C,Y;MAAA,2B;K;+FAEI,Y;MAAQ,qBAAA,kBN/R8C,CM +RxC,CN/RwC,CM+R9C,C;K;gGAEZ,Y;MAAA,4B;K;IAY2B,oG;MAAA,kC;MAAS,uB;K;mJACG,Y;MAAQ,O AAA,kBAAM,O;K;wGACrC,iB;MAAuC,Q;MAAA,eAAA,kBN/SG,CM+SG,KN/SH,CM+SH,mBAAgB,E;K;;qG AJnE,Y;MACI,IAAI,2BAAJ,C;QACI,yH;OAKJ,OAAO,kC;K;4CAGf,Y;MACI,OAA,Y,SAAZ,wBAAY,EAAS,kB AAT,EAAoB,kBAAM,UAAV,GAAqB,kBAAM,MAAN,GAAc,CAAd,IAArB,GAA0C,kBAAM,aAAN,GAAqB,C AArB,IAA1D,EAAkF,wBAAlF,C;K;IArB4B,oE;MAAA,kC;MAA+B,6B;K;mHACHd,Y;MAAQ,OAAA,kBAAM, O;K;IACqC,4E;MAAA,qB;QAAE,yBAAK,EAAL,C;O;K;qEAA5E,Y;MAAiD,OAAqB,OAAb,aAAR,oBAAQ,CA Aa,EAAL,iEAAJ,CAAiB,W;K;wEACvF,iB;MAA4C,Q;MAAA,eAAA,kBNpSU,CMoSJ,KNpSI,CMoSv,YAAoB,o BAAPB,O;K;;IAdxD,uD;MACI,sBAAiB,I;MACjB,YAAY,eAAK,KAAL,C;MACZ,IAAI,aAAJ,C;QAAMB,OAA O,I;MAC1B,YAAY,aAAA,KAAM,MAAN,EAAa,sBAAY,CAAZ,IAAb,C;MAEZ,mE;K;IA8BJ,iD;MAM+B,UAK O,M;MATIC,YAAY,C;MACZ,aAAa,mBAAc,WAAAY,OAA1B,C;MAEb,OAAO,QAAQ,WAAAY,OAA3B,C;QACI, WAAW,wBAAY,YAAZ,EAAY,oBAAZ,Q;QACX,IAAI,SAAQ,EAZ,C;UACI,IAAI,UAAS,WAAAY,OAAzB,C;Y ACI,MAAM,gCAAYB,mCAAzB,C;UAEV,MAAO,gBAAO,wBAAY,cAAZ,EAAY,sBAAZ,UAAP,C;eACJ,IAAI, SAAQ,EAZ,C;UACH,IAAI,UAAS,WAAAY,OAAzB,C;YACI,MAAM,gCAAYB,kCAAzB,C;UAEV,IAAI,uBAA Y,KAAZ,MAAsB,GAA1B,C;YACI,MAAM,gCAAYB,4DAAzB,C;UAEV,IAAI,EAAuB,kBAAK,EAAL,CAAvB,0 CAAY,KAAZ,EAJ,C;YACI,MAAM,gCAAYB,mCAAzB,C;UAEV,eAA2B,eAAZ,WAAAY,EAae,KAaf,EAAsB, KAAM,YAAY,KAAxC,C;UAC3B,iBAAwD,MAAvC,W3GhKmE,W2GgK7C,K3GhK6C,E2GgKtC,Q3GhKsC,C2 GgK5B,C;UAExD,IAAI,cAAc,KAAM,YAAY,KAApC,C;YACI,MAAM,8BAA0B,sBAAmB,UAAAnB,oBAA1B,C;

UAEV,MAAO,gBAAO,KAAM,YAAN,aAakB,UAAIB,CAAP,C;UACP,QAAQ,Q;;UAER,MAAO,gBAAO,IAAP,C;;MAGf,OAAO,MAAO,W;K;IAGlB,2D;MAEI,YAAY,aAAa,CAAb,I;MACZ,iBAAiB,qBAAK,UAAI,IAAmB,E;MAGpC,OAAO,QAAQ,gBAAR,IAAkB,CAAE,kBAAK,EAAL,CAAF,wCAAK,KAAL,EAazB,C;QACI,oBAAoB,CAAC,aAAa,EAAb,IAAD,KAAqB,qBAAK,KAAL,IAAc,EAAnC,K;QACpB,IAAqB,CAAjB,qCAAyB,UAA7B,C;UACI,aAAa,a;UACb,qB;;UAEA,K;;MAGR,OAAO,K;K;I3GxZX,yB;MAQiB,Q;MADb,aAAa,E;MACb,wBAAa,KAAb,gB;QAAa,WAAb,UAAa,KAAb,O;QACI,8BAAU,IAAV,C;;MAEJ,OAAO,M;K;IAGX,yC;MAa+B,Q;MAH3B,IAAI,SAAS,CAAT,IAAc,SAAS,CAAvB,IAA4B,CAAA,KAAM,OAAN,GAAa,MAAb,QAAsB,MAAtD,C;QACI,MAAM,8BAA0B,WAAS,KAAM,OAaf,kBAA+B,MAA/B,kBAAgD,MAA1E,C;MACV,aAAa,E;MACc,gBAS,MAAT,I;MAA3B,iBAAc,MAAd,wB;QACI,8BAAU,MAAM,KAAN,CAAV,C;;MAEJ,OAAO,M;K;IAGX,mC;MAOiB,Q;MADb,aAAa,E;MACb,wBAAa,SAAb,gB;QAAa,WAAb,UAAa,SAAb,O;QACI,8BAAU,IAAV,C;;MAEJ,OAAO,M;K;IAGX,2D;MAY2C,0B;QAAA,aAakB,C;MAAG,wB;QAAA,WAAgB,SAAK,O;MACjF,oCAAa,4BAAmB,UAAAnB,EAA+B,QAA/B,EAAYC,SAAK,OAA9C,C;MACb,aAAa,E;MACb,iBAAc,UAAAd,UAA+B,QAA/B,U;QACI,8BAAU,UAAK,KAAL,CAAV,C;;MAEJ,OAAO,M;K;IASkB,gD;MAAA,qB;QAAE,+CAAI,EAaj,E;O;K;IAN/B,kC;MAMI,OAAO,kBAAU,gBAAV,EAakB,+BAAIB,C;K;IAiBiC,oE;MAAA,qB;QAAE,+CAAI,qBAaA,EAAb,IAAJ,E;O;K;IAd9C,wD;MAYqC,0B;QAAA,aAakB,C;MAAG,wB;QAAA,WAAgB,SAAK,O;MAC3E,oCAAa,4BAAmB,UAAAnB,EAA+B,QAA/B,EAAYC,gBAAzC,C;MACb,OAAO,kBAAU,WAAW,UAAAX,IAAV,EAaiC,2CAAjC,C;K;IAGX,mC;MAQI,OAAO,WAAW,SAAX,EAaiB,CAAjB,EAAoB,gBAApB,EAA0B,KAA1B,C;K;IAGX,mF;MAeI,0B;QAAA,aAakB,C;MACIB,wB;QAAA,WAAgB,SAAK,O;MACrB,sC;QAAA,yBAakC,K;MAEIC,oCAAa,4BAAmB,UAAAnB,EAA+B,QAA/B,EAAYC,SAAK,OAA9C,C;MACb,OAAO,WAAW,SAAX,EAaiB,UAAjB,EAA6B,QAA7B,EAAuC,sBAAvC,C;K;IAGX,sC;MAQI,OAAO,WAAW,SAAX,EAaiB,CAAjB,EAAoB,gBAApB,EAA4B,KAA5B,C;K;IAGX,sF;MAeI,0B;QAAA,aAakB,C;MACIB,wB;QAAA,WAAgB,SAAK,O;MACrB,sC;QAAA,yBAakC,K;MAEIC,oCAAa,4BAAmB,UAAAnB,EAA+B,QAA/B,EAAYC,gBAAzC,C;MACb,OAAO,WAAW,SAAX,EAaiB,UAAjB,EAA6B,QAA7B,EAAuC,sBAAvC,C;K;uFAGX,qB;MAMwD,OAAA,SAAY,c;K;mFAEpE,qB;MAWsD,OAAA,SAAY,c;K;uFAEIE,qB;MAMwD,OAAA,SAAY,c;K;mFAEpE,qB;MAWsD,OAAA,SAAY,c;K;yFAEIE,qC;MACoF,OAAA,SAAY,SAAQ,GAAR,EAAa,SAAb,C;K;iGAehG,qC;MACwF,OAAA,SAAY,aAAY,GAAZ,EAaiB,SAajB,C;K;+FAEpG,kC;MACiF,OAAA,SAAY,YAAW,CAAX,EAAC,QAAAd,C;K;2FAE7F,wB;MACgE,OAAA,SAAY,UAAS,CAAT,C;K;iFAE5E,iC;MACqE,OAAA,SAAY,WAAU,UAAV,C;K;mFAEjF,2C;MACoF,OAAA,SAAY,WAAU,UAAV,EAAsB,QAAtB,C;K;4EAehG,0B;MAGuD,OAAA,SAAY,QAAO,GAAP,C;K;wEAEnE,4B;MAGgE,OAAA,SAAY,OAAM,KAAN,C;K;yFAK5E,2C;MACyF,OAAA,SAAY,SAAQ,OAAR,EAaiB,WAAjB,C;K;IAErG,iD;MAOkD,0B;QAAA,aAAsB,K;MACpE,IAAI,UAAJ,C;QACI,SAAS,SAAK,O;QACd,SAAS,KAAM,O;QACf,UTGG,MAAO,KSHM,ETGN,ESHU,ETGV,C;QSFV,IAAI,QAAO,CAAX,C;UAAc,OAAO,KAak,EAAL,I;QACrB,iBAAc,CAAd,UAAsB,GAAtB,U;UACI,eAAe,qBAAK,KAAL,C;UACf,gBAAgB,iBAAM,KAAN,C;UAEhB,IAAI,aAAY,SAAhB,C;YACI,WAAoB,cAAT,QAAS,C;YACpB,YAAsB,cAAV,SAAU,C;YAEtB,IAAI,aAAY,SAAhB,C;cACwB,kBAAT,Q;cAAX,WDIO2C,gCAAY,cAfrB,YAAY,CAAZ,C;cCkPZ,kBAAV,S;cAAZ,YDnO2C,gCAAY,cAfrB,YAAY,CAAZ,C;cCoPIC,IAAI,aAAY,SAAhB,C;gBACI,OAAgB,iBAAT,QAAS,EAAU,SAAV,C;;QAKhC,OAAO,KAak,EAAL,I;;QAEp,OAAO,4BAAU,KAAV,C;;K;IAIf,4C;MAOqF,oCAakB,KAAIB,C;K;IAErF,wD;MASI,OAAW,UAAJ,GACE,4BAAL,SAAK,EAA4B,KAA5B,CADF,GAGE,kBAAL,SAAK,EAakB,KAAIB,C;K;IAIkD,oD;MAAU,OAAE,UAAF,CAAE,EAAU,CAAV,EAA0B,IAA1B,C;K;;IAIvE,+C;MAAQ,oC;K;2F6G/SZ,oC;MACiF,O7G2Me,kB6G3ME,oBAAH,EAAG,C7G2MF,E6G3Mc,S7G2Md,C;K;mG6GzMhG,oC;MACqF,O7G2Me,sB6G3MM,oBAAH,EAAG,C7G2MN,E6G3MkB,S7G2MIB,C;K;I6GzMpG,mD;MAIoD,0B;QAAA,aAAsB,K;MACTE,IAAI,CAAC,UAAAL,C;QACI,O7GsMqF,qB6GtM7D,M7GsM6D,E6GtMrD,C7GsMqD,C;;Q6GpMrF,OAAO,yBAAc,CAAd,EAaiB,MAAjB,EAAYB,CAAzB,EAA4B,MAAO,OAAAnC,EAA2C,UAA3C,C;K;IAGf,iE;MAIqE,0B;QAAA,aAAsB,K;MACvF,IAAI,CAAC,UAAAL,C;QACI,O7G2LqF,qB6G3L7D,M7G2L6D,E6G3LrD,U7G2LqD,C;;Q6GzLrF,OAAO,yBAAc,UAAAd,EAA0B,MAA1B,EAakC,CAAIC,EAAqC,MAAO,OAA5C,EAAoD,UAApD,C;K;IAGf,iD;MAIkD,0B;QAAA,aAAsB,K;MACpE,IAAI,CAAC,UAAAL,C;QACI,O7GmLoE,mB6GnL9C,M7GmL8C,C;;Q6GjLpE,OAAO,yBAAc,mBAAS,MAAO,OAAhB,IAAd,EAAAsC,MAAtC,EAA8C,CAA9C,EAaiD,MAAO,OAAxD,EAAGe,UAAhE,C;K;IAGf,mC;MAGI,aACa,S7G0L2D,O6G1LhD,K7G0LgD,C;M6GzLxE,OAAO,kBAakB,MAAO,OAAP,KAAe,C;K;IAG5C,4B;MAKoD,gCAAU,C;M

AAV,U;QAAuB,kBAAR,yB;QAAQ,c;;UpH2nDvD,U;UADhB,IAAI,0CAAsB,qBAA1B,C;YAAqC,aAAO,I;YAA P,e;WACrB,+B;UAAhB,OAAgB,gBAAhB,C;YAAgB,2B;YAAM,IAAI,CoH3nD4D,aAAT,qBpH2nDxC,OoH3nD wC,CAAS,CpH2nDhE,C;cAAyB,aAAO,K;cAAP,e;;UAC/C,aAAO,I;;;QoH5nDgE,iB;OAAvB,W;K;IAEpD,gD;M ASiD,0B;QAAA,aAAsB,K;MAOx,C;Q;MAN3B,IAAI,iBAAJ,C;QAAkB,OAAO,a;MACzB,IAAI,aAAJ,C;QAAMb ,OAAO,K;MAC1B,IAAI,CAAC,UAAAL,C;QAAiB,OAAO,kBAAQ,KAAR,C;MAExB,IAAI,SAAK,OAAAL,KAAe, KAAM,OAAzB,C;QAAiC,OAAO,K;MAEb,OAAAL,SAAK,O;MAA3B,iBAAc,CAAd,wB;QACI,eAAe,qBAAK,K AAL,C;QACf,gBAAgB,iBAAM,KAAAN,C;QACHb,IAAI,CAAU,SAAT,QAAS,EAAO,SAAP,EAakB,UAAIB,CA Ad,C;UACI,OAAO,K;;MAIf,OAAO,I;K;IAIX,sF;MACkH,0B;QAAA,aAAsB,K;MACpI,oCAAKB,UAAIB,EAA8 B,KAA9B,EAAqC,WAArC,EAakD,MAAID,EAA0D,UAA1D,C;K;IAGJ,+B;MAYI,OvGmMmD,mBAAS,CuGn M5D,G7GwH4F,oB6GxHzD,C7GwHyD,E6GxHtD,C7GwHsD,CAvC9B,c6GjFrC,G7GqHoD,oB6GrHZ,C7GqHY, C6GrH7E,GAAyE,S;K;IAG7E,iC;MASI,OvGuLmD,mBAAS,CuGvL5D,G7G4G4F,oB6G5GzD,C7G4GyD,E6G5 GtD,C7G4GsD,CAlB9B,c6G1FrC,G7GyGoD,oB6GzGZ,C7GyGY,C6GzG7E,GAAyE,S;K;IAG7E,8B;MAOiB,IA AN,I;MIH/FP,IAAI,E0H8FI,KAAK,C1H9FT,CAAJ,C;QACI,c0H6Fc,oD;Q1H5Fd,MAAM,gCAAyB,OAAQ,WA AjC,C;O0H6FH,QAAM,CAAN,C;aACH,C;UAAK,S;UAAAL,K;aACA,C;UAAU,OAAAL,SAAK,W;UAAV,K;gBAE L,aAAa,E;UACb,IAAI,EvGgKoC,qBAAU,CuGhK9C,CAAJ,C;YACI,QAAQ,SAAK,W;YACb,YAAy,C;YACZ,O AAO,IAAP,C;cACI,IAAI,CAAC,QAAU,CAAX,MAAiB,CAArB,C;gBACI,UAAU,C;eAEd,QAAQ,UAAW,C;cA CnB,IAAI,UAAS,CAAb,C;gBACI,K;eAEJ,KAAK,C;;;UAGb,OAAO,M;;MAnBf,W;K;IAwBJ,4D;MAOqE,0B;QA AA,aAAsB,K;MACvF,O7GkFiG,kB6GIFnF,WAAO,6BAAM,gBAAO,QAAP,CAAb,EAAmC,UAAJ,GAAgB,KA AhB,GAA2B,IAA1D,C7GkFmF,E6GIFIB,6BAAM,iCAAwB,QAAXB,C7GkFY,C;K;I6GhFrG,4D;MAM+D,0B;Q AAA,aAAsB,K;MACjF,O7GyEiG,kB6GzEnF,WAAO,6BAAM,gBAAe,oBAAR,OAAQ,CAAf,CAAb,EAA6C,UA AJ,GAAgB,KAAhB,GAA2B,IAApE,C7GyEmF,E6GzEA,oBAAR,OAAQ,C7GyEA,C;K;I6GvErG,iE;MAC0E,0B; QAAA,aAAsB,K;MAC5F,O7GqEiG,kB6GrEnF,WAAO,6BAAM,gBAAO,QAAP,CAAb,EAAmC,UAAJ,GAAgB, IAAhB,GAA0B,GAAzD,C7GqEmF,E6GrEpB,6BAAM,iCAAwB,QAAXB,C7GqEc,C;K;I6GnErG,iE;MACoE,0B; QAAA,aAAsB,K;MACtF,O7GiEiG,kB6GjEnF,WAAO,6BAAM,gBAAe,oBAAR,OAAQ,CAAf,CAAb,EAA6C,U AAJ,GAAgB,IAAhB,GAA0B,GAAnE,C7GiEmF,E6GjEF,oBAAR,OAAQ,C7GiEE,C;K;I8G7OrG,kD;MAEI,IAAI ,gBAAJ,C;QAAsB,MAAM,6BAAyB,qCAAKC,QAAQ,CAAR,IAAI,CAAzB,C;MAC5B,OAAO,CAAC,IAAD,I; K;IAGX,iF;MAQI,IAAI,EAAS,KAAT,oBAAiB,KAAjB,KAA2B,SAAS,QAAXC,C;QACI,OAAO,UAAU,CAAV,E AAa,KAAb,EAAoB,gBAApB,C;OAEX,UAAU,kBAAO,KAAP,C5GwBgC,I;M4GvB1C,IAAI,EAAQ,KAAR,kBA AgB,KAAhB,CAAJ,C;QACI,OAAO,UAAU,CAAV,EAAa,KAAb,EAAoB,gBAApB,C;OAEX,OAAO,SAAW,CA AC,OAAS,IAAV,KAAqB,EAAhC,IAAwC,MAAQ,I;K;IAG3D,yE;MAQI,IAAI,SAAU,EAAV,MAAKB,CAIB,IA AuB,SAAS,QAAP,C,C;QACI,OAAO,UAAU,CAAV,EAAa,KAAb,EAAoB,gBAApB,C;OAEX,YAAy,KAAa,CAA P,KAAO,C;MACzB,IAAI,SAAU,GAAV,MAAKB,GAAtB,C;QACI,OAAO,UAAU,CAAV,EAAa,KAAb,EAAoB,g BAApB,C;OAEX,OAAQ,SAAU,CAAX,GAAkB,KAAIB,GAA4B,I;K;IAGvC,yE;MASI,IAAI,SAAS,QAAb,C;QA CI,OAAO,UAAU,CAAV,EAAa,KAAb,EAAoB,gBAApB,C;OAGX,YAAy,KAAa,CAAP,KAAO,C;MACzB,IAAI ,SAAU,EAAV,MAAiB,CAArB,C;QACI,IAAI,SAAU,GAAV,MAAKB,GAAtB,C;UAEI,OAAO,UAAU,CAAV,EA Aa,KAAb,EAAoB,gBAApB,C;gBAER,IAAI,SAAU,EAAV,MAAiB,EAAR,C;QACH,IAAI,SAAU,GAAV,MAAK B,GAAtB,C;UAEI,OAAO,UAAU,CAAV,EAAa,KAAb,EAAoB,gBAApB,C;gBAER,IAAI,SAAU,GAAV,MAAKB ,GAAtB,C;QACH,OAAO,UAAU,CAAV,EAAa,KAAb,EAAoB,gBAApB,C;OAGX,IAAI,SAAQ,CAAR,UAAa,Q AAjB,C;QACI,OAAO,UAAU,CAAV,EAAa,KAAb,EAAoB,gBAApB,C;OAEX,YAAy,KAAiB,CAAX,QAAQ,C AAR,IAAW,C;MAC7B,IAAI,SAAU,GAAV,MAAKB,GAAtB,C;QACI,OAAO,UAAU,CAAV,EAAa,KAAb,EAAo B,gBAApB,C;OAGX,OAAQ,SAAU,EAAX,GAAoB,SAAU,CAA9B,GAAqC,KAAR,C,GAA+C,O;K;IAG1D,yE;M ASI,IAAI,SAAS,QAAb,C;QACI,UAAU,CAAV,EAAa,KAAb,EAAoB,gBAApB,C;OAGJ,YAAy,KAAa,CAAP,K AAO,C;MACzB,IAAI,SAAU,EAAV,MAAiB,CAArB,C;QACI,IAAI,SAAU,GAAV,KAAkB,GAAtB,C;UAEI,OA AO,UAAU,CAAV,EAAa,KAAb,EAAoB,gBAApB,C;gBAER,IAAI,SAAU,EAAV,MAAiB,CAArB,C;QACH,IAA I,SAAU,GAAV,MAAKB,GAAtB,C;UAEI,OAAO,UAAU,CAAV,EAAa,KAAb,EAAoB,gBAApB,C;gBAER,IAAI, SAAU,EAAV,IAAgB,CAAP,C;QACH,OAAO,UAAU,CAAV,EAAa,KAAb,EAAoB,gBAApB,C;aACJ,IAAI,SA AU,GAAV,MAAKB,GAAtB,C;QACH,OAAO,UAAU,CAAV,EAAa,KAAb,EAAoB,gBAApB,C;OAGX,IAAI,SA AQ,CAAR,UAAa,QAajB,C;QACI,OAAO,UAAU,CAAV,EAAa,KAAb,EAAoB,gBAApB,C;OAEX,YAAy,KAAi

B,CAAX,QAAQ,CAAR,IAAW,C;MAC7B,IAAI,SAAU,GAAV,MAAkB,GAAtB,C;QACI,OAAO,UAAU,CAAV,EA  
EAa,KAAb,EAAoB,gBAApB,C;OAGX,IAAI,SAAQ,CAAR,UAAa,QAAjB,C;QACI,OAAO,UAAU,CAAV,EA  
Aa,KAAb,EAAoB,gBAApB,C;OAEX,YAAy,KAAiB,CAAX,QAAQ,CAAR,IAAW,C;MAC7B,IAAI,SAAU,GAA  
V,MAAkB,GAAtB,C;QACI,OAAO,UAAU,CAAV,EAa,KAAb,EAAoB,gBAApB,C;OAEX,OAAQ,SAAU,EA  
X,GAAoB,SAAU,EAA9B,GAAuC,SAAU,CAAjD,GAAwD,KAAxD,GAaKE,O;K;;;IAmB7E,oE;MAkB0B,UAGJ,  
MAHI,EAKJ,MALI,EAMJ,MANI,EASJ,MATI,EAUJ,MAVI,EAWJ,MAXI,EAgBA,MahBA,EAiBA,MAjBA,EA  
kBA,MAIBA,EAoBA,MApBA,EAqBA,OArBA,EASBA,OAiBA,EAuBA,O;M3H9JtB,IAAI,E2HgII,cAAc,CAAd,I  
AAmB,YAAy,MAAO,OAAtC,IAAgD,cAAc,Q3HhIIE,CAAJ,C;QACI,cAda,qB;QAeb,MAAM,gCAAyB,OAAQ,  
WAAjC,C;O2HgIV,YAAy,cAAU,CAAC,WAAW,UAAx,IAAD,IAA0B,CAA1B,IAAV,C;MACZ,gBAAgB,C;M  
AChB,gBAAgB,U;MAEhB,OAAO,YAAy,QAAnB,C;QACI,WAAW,mBAAO,gBAAP,EAAO,wBAAP,Q5G1H2  
B,I;Q4G4HIC,WAAO,GAAP,C;UACI,MAAM,kBAAN,EAAM,0BAAN,YAA0B,OAAL,IAAK,C;eAC9B,WAAO,  
IAAP,C;UACI,MAAM,kBAAN,EAAM,0BAAN,YAA4C,OAAR,QAAS,CAAV,GAAgB,GAAM,C;UAC5C,MA  
AM,kBAAN,EAAM,0BAAN,YAA+C,OAAXB,OAAS,EAAV,GAAMB,GAAM,C;eAEnD,WAAO,KAAP,IAAiB,  
QAAQ,KAAzB,C;UACI,MAAM,kBAAN,EAAM,0BAAN,YAA6C,OAAtB,QAAS,EAAV,GAAiB,GAAM,C;UA  
C7C,MAAM,kBAAN,EAAM,0BAAN,YAAuD,OAAB,QAAS,CAAV,GAAiB,EAIB,GAA2B,GAAM,C;UACvD  
,MAAM,kBAAN,EAAM,0BAAN,YAA+C,OAAXB,OAAS,EAAV,GAAMB,GAAM,C;;UAG/C,gBAAgB,uBAAu  
B,MAAvB,EAA+B,IAA/B,EAaQc,SAARc,EAAGD,QAaHd,EAA0D,gBAA1D,C;UAChB,IAAI,aAAa,CAAjB,C;  
YACI,MAAM,kBAAN,EAAM,0BAAN,YAAqB,0BAA0B,CAA1B,C;YACrB,MAAM,kBAAN,EAAM,0BAAN,Y  
AAqB,0BAA0B,CAA1B,C;YACrB,MAAM,kBAAN,EAAM,0BAAN,YAAqB,0BAA0B,CAA1B,C;;YAErB,MAA  
M,kBAAN,EAAM,0BAAN,YAAkD,OAAB3B,aAAc,EAaf,GAASB,GAAM,C;YACID,MAAM,mBAAN,EAAM,2B  
AAN,aAA6D,OAARc,aAAc,EAaf,GAAuB,EAAXB,GAAiC,GAAM,C;YAC7D,MAAM,mBAAN,EAAM,2BAAN,  
aAA4D,OAAPc,aAAc,CAAF,GAASB,EAavB,GAAgC,GAAM,C;YAC5D,MAAM,mBAAN,EAAM,2BAAN,aAA  
oD,OAAB7B,YAAc,EAaf,GAAwB,GAAM,C;YACpD,6B;;;MAMhB,OAAW,KAAM,OAAN,KAAc,SAAB,GAA  
6B,KAA7B,GAA8C,UAAAN,KAAM,EAAO,SAAP,C;K;;IAQzD,mE;MAiByB,Q;M3H9LrB,IAAI,E2HwLI,cAAc,C  
AAAd,IAAmB,YAAy,KAAM,OAARc,IAA6C,cAAc,Q3HxL/D,CAAJ,C;QACI,cAda,qB;QAeb,MAAM,gCAAyB,O  
AAQ,WAAjC,C;O2HwLV,gBAAgB,U;MAChB,oBAAoB,sB;MAEpB,OAAO,YAAy,QAAnB,C;QACI,WAAW,K  
AAmB,CAAb,gBAAa,EAAb,wBAAa,O;QAE1B,YAAQ,CAAR,C;UACI,aAAc,gBAAy,OAAL,IAAK,CAAZ,C;a  
ACIB,YAAS,CAAT,KAAc,EAAd,C;UACI,WAAW,eAAe,KAAf,EAASB,IAAtB,EAA4B,SAASB,EAuB,C,QAavC  
,EAAiD,gBAAjD,C;UACX,IAAI,QAAQ,CAAZ,C;YACI,aAAc,gBAAO,gBAAP,C;YACd,yBAAa,CAAC,IAAD,I  
AAb,K;;YAEA,aAAc,gBAAy,OAAL,IAAK,CAAZ,C;YACd,wBAAa,CAAb,I;;eAGR,YAAS,CAAT,KAAc,EAAd  
,C;UACI,aAAW,eAAe,KAAf,EAASB,IAAtB,EAA4B,SAASB,EAuB,C,QAavC,EAAiD,gBAAjD,C;UACX,IAAI,  
UAAQ,CAAZ,C;YACI,aAAc,gBAAO,gBAAP,C;YACd,yBAAa,CAAC,MAAD,IAAb,K;;YAEA,aAAc,gBAAy,O  
AAL,MAAK,CAAZ,C;YACd,wBAAa,CAAb,I;;eAGR,YAAS,CAAT,KAAc,EAAd,C;UACI,aAAW,eAAe,KAAf,E  
AASB,IAAtB,EAA4B,SAASB,EAuB,C,QAavC,EAAiD,gBAAjD,C;UACX,IAAI,UAAQ,CAAZ,C;YACI,aAAc,gB  
AAO,gBAAP,C;YACd,yBAAa,CAAC,MAAD,IAAb,K;;YAEA,WAAy,MAAD,GAAQ,KAAR,IAAQB,EAARB,G  
AA2B,K;YACtC,UAAW,SAAS,IAAV,GAAoB,K;YAC9B,aAAc,gBAAy,OAAL,IAAK,CAAZ,C;YACd,aAAc,gB  
AAW,OAAL,GAAL,CAAX,C;YACd,wBAAa,CAAb,I;;;UAIJ,UAAU,CAAV,EAa,SAAb,EAawB,gBAaxB,C;U  
ACA,aAAc,gBAAO,gBAAP,C;;;MAK1B,OAAO,aAAc,W;K;ICtQzB,uC;MAU2D,OAAwB,CAAxB,2BAawB,mB  
AAS,SAAT,C;K;IAEnF,oC;MAKI,OAAQ,OAAW,mBAAL,SAAK,CAAX,C;K;IAGZ,6C;MAMI,IAAI,cAAS,SA  
Ab,C;QACI,iBAASB,SAAY,Y;QACIC,IAAI,kBAAJ,C;UACS,SAAL,eAA+B,iBAAc,SAAd,E;;UAE/B,UAAW,W  
AAI,SAAJ,C;;Q;IAUnB,6C;MAC4B,UAAjB,M;MAAP,OAAO,WAAiB,OAAL,SAAY,YAAjB,4CAA+D,W;K;IAI  
9E,iC;MACI,gBAAqB,sB;MACrB,iBAASB,E;MACtB,kBAA+B,E;MAC/B,uBAAiC,C;K;uDAEjC,qB;MACc,qBA  
AV,SAAU,EAAC,EAAD,EAakB,EAIB,C;MACV,OAAO,aAAO,W;K;gDAGIB,qB;MAA6D,gBAAR,c;MAAQ,c;  
;Q1I41Y7C,Q;QAaHb,wBAAgB,SAaHb,gB;UAAgB,cAAA,SAaHb,M;UAAASB,IAAc,00I51Y+B,c1I41Y7C,C;Y  
AAwB,aAAO,I;YAAP,e;;QAC9C,aAAO,K;;;M0I71Y8C,iB;K;sDAErD,wC;MACI,KAAK,qBAAL,SAAK,EAAC,  
MAAd,EAASB,SAATB,CAAL,C;QAAyC,M;MAEzC,YAAy,SAAK,M;MACjB,OAAO,aAAP,C;QACI,KAAM,qB  
AAN,KAAM,EAAC,MAAd,EAASB,aAAAtB,CAAN,C;UAA8C,M;QAC9C,QAAQ,KAAM,M;;K;sDAItB,wC;MAS  
gB,IAAiB,IAAjB,EA2BE,M;MANCd,aAAO,gBAAO,MAAP,CAAE,gBAAO,SAAP,C;MACTB,gBAAgB,SAAK,W

;MACrB,IAAI,eAAQ,SAAR,CAAJ,C;QACI,aAAO,gBAAO,kCAAP,CAA2C,gBAAO,SAAP,CAAkB,gBAAO,KA  
AP,C;QACpE,OAAO,K;OAEH,cAAY,MAAK,SAAL,C;MAEpB,YAAY,CAAiB,OAAZ,SAAY,MAAjB,2D;MAC  
Z,IAAI,aAAJ,C;QzHyBG,SyHxBwB,WAAN,KAAM,EAAQ,SAAR,C;QAAvB,iBAAoD,KAACK,CAAT,GAAY,C  
AAZ,GAAMB,KAAe,gBAAf,I;QACnE,IAAI,eAAc,CAAIB,C;UAAqB,aAAO,gBAAO,SAAP,CAAkB,gBAAO,IA  
AP,C;QAC9C,IAAI,ezG8MoC,YAAU,CyG9MID,C;UACI,kBAAW,K;UACX,uBAAgB,U;;UAEhB,QAAQ,wBAA  
iB,KAAjB,EAAwB,UAAxB,C;;QAEZ,IAAI,MzGgNuC,UAAAS,CyGhNpD,C;UAEuB,U;UAAA,IAAI,eAAc,CAA  
B,C;YAAA,SAAqB,C;;Y1Gq+BpC,U;YADhB,YAAY,C;YACI,oB0Gr+B+C,S1Gq+B/C,C;YAAhB,OAAgB,gBA  
AhB,C;cAAGB,sC;cAAM,I0Gr+BgE,U1Gq+BiD,oB0Gr+BkD,MAAK,E1Gq+BrE,C;gBAAwB,qB;;Y0Gr+Bf,SA  
4B,I1Gs+BpD,K0Gt+BoD,I;;UAA/C,yB;U5GorCC,kB;UADb,YAAY,C;UACC,S4GmrCK,aAAN,KAAM,C5GmrC  
L,W;UAAb,OAAa,gBAAb,C;YAAa,wB;Y4GlrCG,I5GkrCU,oBAAMB,cAAnB,EAAmB,sBAAnB,U4GlrCN,gBA  
AJ,C;cAA2B,aAAO,uB;YACIC,aAAO,gB5GirCgC,I4GjrChC,CAAa,gBAAO,IAAP,C;;;UAGxB,aAAO,gBAAO,K  
AAP,CAAc,gBAAO,IAAP,C;;;QAGzB,aAAO,gBAAO,SAAP,CAAkB,gBAAO,IAAP,C;;;MAG7B,iBAAiB,mC;M  
ACjB,IpIuHoD,CoIvHhD,UpIuHiD,UoIvHrD,C;QACI,uBAAuB,SAAS,M;QACtB,8B;QAAV,OAAU,gBAAV,C;U  
AAU,qB;UACJ,qBAAF,CAAE,EAAc,gBAAd,EAAgC,cAAhC,C;;OAGV,OAAO,I;K;yDAGX,6B;MAIwB,Q;MA  
HpB,mBAAwB,C;MACxB,gBAAqB,C;MACrB,mBAAwB,C;MACJ,OxHyIjB,MAAO,KwHzIgB,eAAS,OAAT,G  
AAkB,oBAAlB,IxHyIhB,EwHzIiD,KAAM,OAAN,GAAe,UAAf,IxHyIjD,C;MwHzIV,eAAY,CAAZ,oB;QACI,QA  
AQ,iBAAY,iBAAN,KAAM,CAAN,GAACK,GAAIB,IAAN,C;QACR,IAAI,MAAK,2BAACK,iBAAT,eAAS,CAA  
T,GAAqB,GAARb,IAAT,CAAT,C;UAA6C,K;QAC7C,IAAI,MAAK,EAAT,C;UACI,8BAAGb,CAAhB,I;UACA,e  
AAe,S;UAcF,YAAY,G;;MAGpB,IAAI,gBAAGb,CAApB,C;QAAuB,OAAO,K;MAC9B,OAAO,eAAe,CAAf,IAA  
oB,iBAAY,iBAAN,KAAM,CAAN,IAAmB,YAAnB,GAAkC,CAAIC,KAAN,MAA+C,EAAIE,C;QACI,8BAAGb,  
CAAhB,I;MAGJ,OAAa,YAAN,KAAM,EAAS,YAAT,CAAN,IAA+B,cAAW,eAAe,CAAf,IAAX,uCAA/B,C;K;;y  
HC/H+C,Y;MAAQ,W;K;IAEtE,gD;MACKB,UAMP,M;MANO,IAAI,aAAY,CAAhB,C;QACV,Y;;QAEA,UxBsY8  
C,MAAW,KwBtY/C,IxBsY+C,EwBtYtC,QxBsYsC,C;QwBrYzD,OAAA,IAAO,OxB2UmC,MAAW,KwB3UpC,K  
xB2UoC,CwB3UxC,GAAa,GAAnB,CAAP,GAAiC,GAAjC,GxBwV2C,MAAW,MwBxVV,KxBwVU,C;;MwB5V  
1D,kB;MAMO,IxByUuC,MAAW,KwBzU1C,OxByU0C,CwBzU9C,GAAe,MAAnB,C;QAEmC,SAA9B,OAA,Y,S  
AAQ,QAAR,C;;QAGpB,exBoU0C,MAAW,KwBpUIC,OxBoUkC,C;QwBnUrD,qBAA8B,QAA,Y,axBgRC,MAA  
W,MAvCV,MAAW,OwBzOU,QxByOV,CAuCD,CwBhRA,GAAwB,QAAPc,C;QAC1C,SAAI,UAAU,CAAd,GA  
AiB,MAAG,cAApB,GAAY,c,c;;MAP7C,a;K;IAWJ,6C;MACI,OAAa,KAAY,gBAAe,OAAf,EAAwB,MAAK,4BA  
A2B,QAA3B,CAAL,EAAxB,C;K;ICtBQ,4C;MAFrC,e;MAEsC,0B;MAFtC,iB;MAAA,uB;K;IAAA,mC;MAAA,sC  
;O;MAGI,uEAGY,GAHZ,C;MAIA,yEAGa,MAHb,C;MAIA,yEAGa,SAHb,C;MAIA,+DAGQ,KAHR,C;MAIA,+D  
AGQ,MAHR,C;MAIA,2DAGM,MAHN,C;MAIA,yDAGK,OAHL,C;K;;IAxBA,gD;MAAA,yB;MAAA,wC;K;;IAI  
A,iD;MAAA,yB;MAAA,yC;K;;IAIA,iD;MAAA,yB;MAAA,yC;K;;IAIA,4C;MAAA,yB;MAAA,oC;K;;IAIA,4C;M  
AAA,yB;MAAA,oC;K;;IAIA,0C;MAAA,yB;MAAA,kC;K;;IAIA,yC;MAAA,yB;MAAA,iC;K;;IA3BJ,+B;MAAA,  
4Q;K;;IAAA,oC;MAAA,a;aAAA,a;UAAA,6C;aAAA,c;UAAA,8C;aAAA,c;UAAA,8C;aAAA,S;UAAA,yC;aAAA,  
S;UAAA,yC;aAAA,O;UAAA,uC;aAAA,M;UAAA,sC;gBAAA,6D;;K;;IAiCA,4D;MAGW,Q;MADP,0BAA2C,iB  
AAjB,UAAW,cAAM,EAAU,UAAW,cAArB,C;MAEvC,0BAAsB,CAAtB,C;QAA2B,gBAAS,UAAW,cAAX,GAAMB,UAAW,cAAvC,C;;  
QACnB,Y;MAHZ,W;K;IAOJ,oE;MAGW,Q;MADP,0BAA2C,iBAAjB,UAAW,cAAM,EAAU,UAAW,cAArB,C;M  
AEvC,0BAAsB,CAAtB,C;QAA2B,sBAA8C,uBAArC,UAAW,cAAX,GAAMB,UAAW,cAAO,CAA9C,C;WAC3B  
,0BAAsB,CAAtB,C;QAA2B,iBAA8C,uBAArC,UAAW,cAAX,GAAMB,UAAW,cAAO,CAA9C,C;;QACnB,Y;M  
AHZ,W;K;IAOJ,8D;MAGW,Q;MADP,0BAA2C,iBAAjB,UAAW,cAAM,EAAU,UAAW,cAArB,C;MAEvC,0BA  
AsB,CAAtB,C;QACI,YAAkD,uBAArC,UAAW,cAAX,GAAMB,UAAW,cAAO,C;QACID,aAAa,eAAQ,KAAR,C;  
QAET,sBAAS,KAAT,GAACK,KAAIB,E;UAA2B,a;aAC3B,uBAAQ,CAAR,C;;;aAIR,0BAAsB,CAAtB,C;QAA2  
B,iBAA8C,uBAArC,UAAW,cAAX,GAAMB,UAAW,cAAO,CAA9C,C;;QACnB,Y;MAXZ,W;K;ICrDJ,+B;MAA  
A,mC;MAUuB,wB;MALf,aAAR,OAAO,OAAQ,KAAI,WAA,Y,IAAG,OAAO,SAAX,IAAwB,CAAC,CAAC,OAA  
O,SAAS,K;MADpE,sBAGQ,MAHR,GAIQ,iBAAa,OAAb,CAJR,GAMQ,qBAAW,OAA,X,IAAA,4GACO,+B;K;4  
CAIf,Y;MAAmC,OAAA,mBAAa,U;K;;;IAfpD,2C;MAAA,0C;QAAA,yB;OAAA,mC;K;IAwB2B,+B;MAAC,sB;K  
;IAEW,+D;MAAA,0C;MAAS,mB;MACxC,iBAAgB,yBAAQ,S;K;8DACxB,Y;M5HyEG,Q4HxEC,8BAAQ,QAA

O,cAAP,C;MAAyB,c7IZIC,EAAI,CAAJ,C;M6IY2C,Y7IuF3C,EAAI,CAAJ,C;M6IvFC,OAA4D,aAAR,OAAQ,qC  
AAR,aAAiD,aAN,KAAM,yCAAJD,C;K;;qCAH5D,Y;MAAmC,md;K;sCAMnC,Y;MAAkC,qC;K;;IAKF,4C;M  
AAiC,4E;MAAhC,8B;K;2CACjC,Y;MAA8B,OAAA,gBAAY,M;K;+CAC1C,Y;MAAkC,2C;K;;IAGtC,6B;MAAA  
iC;MAEoC,4E;K;uCACHC,Y;MAA8B,OAAe,U;K;2CAC7C,Y;MAAkC,+B;K;;IAJtC,yC;MAAA,wC;QAAA,uB;  
OAAA,iC;K;IC1CA,gD;MAQ+B,kBAAPB,wBAAC,IAAd,C;MAA0B,I7HgEjC,a;M6HhEA,O7HiEO,W;K;I6H9D  
X,gD;MAQqD,kBAA1B,gBAAhB,sCAAgB,EAAc,IAAd,EAAoB,IAApB,C;MAAiC,sB7HoEID,W6HpEkD,C;MA  
AxD,O7HqEO,W;K;I8HzFX,yC;MAEKD,8B;MAAA,OCGN,aDHwB,yBAAa,QAAb,mCCGxB,C/G+xBgC,sB;K;I  
8GhyB5E,2C;M/IggIW,kBAAY,gB;MAoGH,Q;MAAhB,wB+I7IqB,U/I6Irb,gB;QAAgB,c+I7IIK,U/I6Irb,M;QA  
AsB,IAAI,C+I7IIkB,sB/I6IIP,O+I7IIO,C/I6IItB,C;UAAyB,WAAy,WAAI,OAAJ,C;;M+I7II3D,qB/I8IIO,W;M+I7II  
P,IzIgNwD,CyIhNpD,czIgNqD,UyIhNzD,C;Q9GgKuC,U;Q8G/JnC,qB9G+JyD,OAAtB,+B8G/Jd,mB9G+Jc,uBAA  
sB,CAAO,W;QsGkO7C,kBAAhB,sB;QQ/XC,0C;QACA,IAAI,E9G8QoC,0BAAU,C8G9Q9C,CAAJ,C;UACI,2BA  
AO,GAAP,C;SAEW,sCAAa,GAAb,C;QALnB,sB9H4DG,WsHoUqC,W;QQzXxC,OAAO,I;OAGX,OAAO,K;K;IA  
GX,8C;MAOmB,c;;Q/Ii3YC,Q;QAAhB,wB+Ij3YI,U/Ii3YJ,gB;UAAgB,c+Ij3YZ,U/Ii3YJ,M;UAAsB,I+Ij3YD,sB/I  
i3Ye,O+Ij3Yf,C/Ii3YC,C;YAAwB,aAAO,I;YAAP,e;;QAC9C,aAAO,K;;M+II3YP,e;QACI,kBAA6B,MAAX,UAA  
W,C;Q9GyIM,U;Q8GxIb,a9GwImC,OAAtB,+B8GxIvB,mB9GwIuB,uBAAAsB,CAAO,W;Q8GxIX,kBC/BjB,aD+B  
D,MC/BC,C/Gg1C6C,uBAAzB,CAAyB,C;QbnmB9E,kBAAS,gB;QA2FA,U;QAAA,+B;QAAhB,OAAgB,gBAAh  
B,C;UAAgB,6B;UAAM,I2HzyB4C,4B3HyyB9B,S2HzyB8B,C3HyyB5C,C;YAAwB,WAAy,WAAI,SAAJ,C;;Q2  
HzyBtD,sBAAMf,e3H0yBhF,W2H1yBgF,EAAa,GAAb,C;QACnF,OAAO,I;OAGX,OAAO,K;K;IEnCP,iC;MAAQ  
,8BAAY,IAAK,UAAjB,IAA8B,uBAAy,IAAK,mB;K;IAOvD,oC;MAAQ,8BAAY,IAAK,a;K;ICZ7B,4B;MAGI,O  
AAO,yBAAP,C;QACI,sBAAY,mCAAZ,C;;K;IAIR,uC;MAOI,sBAAY,sCAAgB,gBAAe,IAAf,CAA5B,C;MACA,  
OAAO,S;K;ICbP,4B;MAAQ,mB;K;IACR,mC;MACI,eAAO,K;K;IAKX,4B;MAAQ,mB;K;IACR,mC;MACI,eAA  
O,K;K;iHCoBf,sJ;MAEyC,qB;QAAA,QAakB,I;MAAM,qB;QAAA,QAakB,I;MAAM,uB;QAAA,UAAoB,K;MA  
AO,yB;QAAA,YAAsB,I;MAAM,kC;QAAA,qBAA+B,I;MAAM,qC;QAAA,wBAAkC,K;MAAO,+C;QAAA,kCA  
A4C,K;MAAO,4C;QAAA,+BAAYC,K;MACtT,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,OAAF,IA  
Aa,K;MACb,EAAE,SAAF,IAAe,O;MACf,EAAE,WAAF,IAAiB,S;MACjB,EAAE,oBAAF,IAA0B,kB;MAC1B,E  
AAE,uBAAF,IAA6B,qB;MAC7B,EAAE,iCAAF,IAAuC,+B;MACvC,EAAE,8BAAF,IAAoC,4B;MACpC,OAAO,  
C;K;+GAw0BX,wD;MAEwC,6B;QAAA,gBAAYB,E;MAAI,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;  
MAAO,wB;QAAA,WAAqB,K;MAC/I,QAAQ,E;MACR,EAAE,eAAF,IAAqB,a;MACrB,EAAE,SAAF,IAAe,O;M  
ACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MAChB,OAAO,C;K;6EA6CX,4B;MAE6D,iBAAY  
,KAAZ,C;K;6EAE7D,mC;MAEoE,UAAy,KAAZ,IAAqB,K;K;6EAuBzF,4B;MAE8D,iBAAY,KAAZ,C;K;6EAE9  
D,mC;MAEqE,UAAy,KAAZ,IAAqB,K;K;6EAuB1F,4B;MAEqE,iBAAY,KAAZ,C;K;6EAErE,mC;MAE4E,UAA  
y,KAAZ,IAAqB,K;K;6EAuBjG,4B;MAE+D,iBAAY,KAAZ,C;K;6EAE/D,mC;MAEsE,UAAy,KAAZ,IAAqB,K;  
K;6EAuB3F,4B;MAEGE,iBAAY,KAAZ,C;K;6EAEhE,mC;MAEuE,UAAy,KAAZ,IAAqB,K;K;6EAuB5F,4B;MA  
E6D,iBAAY,KAAZ,C;K;6EAE7D,mC;MAEoE,UAAy,KAAZ,IAAqB,K;K;6EAuBzF,4B;MAE8D,iBAAY,KAAZ,  
C;K;6EAE9D,mC;MAEqE,UAAy,KAAZ,IAAqB,K;K;6EAuB1F,4B;MAEiE,iBAAY,KAAZ,C;K;6EAEjE,mC;M  
AEwE,UAAy,KAAZ,IAAqB,K;K;6EAuB7F,4B;MAEKI,iBAAY,KAAZ,C;K;6EAEIE,mC;MAEyE,UAAy,KAAZ  
,IAAqB,K;K;6GC3oC9F,wD;MAEqC,6B;QAAA,gBAA+B,I;MAAM,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aA  
AuB,K;MAAO,wB;QAAA,WAAqB,K;MACpJ,QAAQ,E;MACR,EAAE,eAAF,IAAqB,a;MACrB,EAAE,SAAF,IA  
Ae,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MAChB,OAAO,C;K;miAiCX,+B;MAEGD,  
mC;QAAA,sBAAgC,K;MAC5E,QAAQ,E;MACR,EAAE,qBAAF,IAA2B,mB;MAC3B,OAAO,C;K;4EC9CX,4B;  
MAEGE,iBAAY,KAAZ,C;K;4EAgChE,4B;MAEyE,iBAAY,KAAZ,C;K;4EAiBzE,4B;MAEmE,iBAAY,KAAZ,C;  
K;4EAyYnE,4B;MAE0E,iBAAY,KAAZ,C;K;oIC7a1E,4H;MAE8C,qB;QAAA,QAAiB,E;MAAI,6B;QAAA,gBAA  
gC,E;MAAW,iC;QAAA,oBAA2D,E;MAAW,iC;QAAA,oBAA2D,E;MAAW,qC;QAAA,wBAmJvJ,U;OAnJqO,+B  
;QAAA,kBAmJrO,U;OAnJ6S,4B;QAAA,eAA+B,S;MAC3a,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAA  
E,eAAF,IAAqB,a;MACrB,EAAE,mBAAF,IAAyB,iB;MACzB,EAAE,mBAAF,IAAyB,iB;MACzB,EAAE,uBAAF,  
IAA6B,qB;MAC7B,EAAE,iBAAF,IAAuB,e;MACvB,EAAE,cAAF,IAAoB,Y;MACpB,OAAO,C;K;wiAYX,mC;M  
AEgD,2B;QAAA,cAAuB,E;MAAI,0B;QAAA,aAAsB,E;MAC7F,QAAQ,E;MACR,EAAE,aAAF,IAAmB,W;MAC  
nB,EAAE,YAAF,IAAkB,U;MACIB,OAAO,C;K;8HAKEX,+D;MAEqG,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,

aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MAC/K,QAAQ,E;MACR,EAAE,aAAF,IAAmB,W;MACnB,EAAE,SA  
AF,IAAe,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MACH  
B,OAAO,C;K;4HAwBX,iE;MAE0C,4B;QAAA,eAAwB,E;MAAI,wB;QAAA,WAAyB,I;MAAM,uB;QAAA,UAA  
oB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MAC/K,QAAQ,E;MACR,EAAE,cAAF,IAAO  
B,Y;MACpB,EAAE,UAAF,IAAgB,Q;MACHB,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EA  
AE,UAAF,IAAgB,Q;MACHB,OAAO,C;K;sGAUqE,qB;MAAQ,OAAW,U;K;sGAEnB,qB;MAAQ,OAAW,U;K;4G  
AEhB,qB;MAAQ,OAAc,a;K;wGAS1B,qB;MAAQ,OAAy,W;K;0HAEX,qB;MAAQ,OAAqB,oB;K;kGASnD,qB;  
MAAQ,OAAAS,Q;K;oGAehB,qB;MAAQ,OAAU,S;K;sGAejB,qB;MAAQ,OAAW,U;K;wHAEV,qB;MAAQ,OAA  
oB,mB;K;wHAE5B,qB;MAAQ,OAAoB,mB;K;kHAE/B,qB;MAAQ,OAAiB,gB;K;kHAEzB,qB;MAAQ,OAAiB,g  
B;K;oHASd,qB;MAAQ,OAAkB,iB;K;oHAE1B,qB;MAAQ,OAAkB,iB;K;oHAE1B,qB;MAAQ,OAAkB,iB;K;wIA  
EhB,qB;MAAQ,OAA4B,2B;K;4FC1MnI,uD;MAE8B,oB;QAAA,OAAgB,I;MAAM,sB;QAAA,SAAe,C;MAAG,u  
B;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACHJ,QAAQ,E;MACR,EAA  
E,MAAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MA  
CIB,EAAE,UAAF,IAAgB,Q;MACHB,OAAO,C;K;kGAuBX,sE;MAEiC,6B;QAAA,gBAA8B,I;MAAM,oB;QAAA,  
OAAgB,I;MAAM,sB;QAAA,SAAe,C;MAAG,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;Q  
AAA,WAAqB,K;MACvL,QAAQ,E;MACR,EAAE,eAAF,IAAqB,a;MACrB,EAAE,MAAF,IAAY,I;MACZ,EAAE,  
QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;M  
ACHB,OAAO,C;K;kGA8DX,8U;MAEiC,uB;QAAA,UAAgB,C;MAAG,uB;QAAA,UAAgB,C;MAAG,uB;QAAA,  
UAAgB,C;MAAG,uB;QAAA,UAAgB,C;MAAG,sB;QAAA,SAAiB,C;MAAG,uB;QAAA,UAAkB,C;MAAG,6B;Q  
AAA,gBAA8B,I;MAAM,sB;QAAA,SAAkB,I;MAAM,uB;QAAA,UAAoB,K;MAAO,wB;QAAA,WAAqB,K;MA  
AO,sB;QAAA,SAAmB,K;MAAO,uB;QAAA,UAAoB,K;MAAO,gC;QAAA,mBAA6B,K;MAAO,gC;QAAA,mBA  
A6B,K;MAAO,0B;QAAA,aAAuB,K;MAAO,8B;QAAA,iBAA2B,K;MAAO,6B;QAAA,gBAA0B,K;MAAO,+B;Q  
AAA,kBAA4B,K;MAAO,kC;QAAA,qBAA+B,K;MAAO,6B;QAAA,gBAA0B,K;MAAO,8B;QAAA,iBAA2B,K;  
MAAO,kC;QAAA,qBAA+B,K;MAAO,oB;QAAA,OAAgB,I;MAAM,sB;QAAA,SAAe,C;MAAG,uB;QAAA,UAA  
oB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MAC3wB,QAAQ,E;MACR,EAAE,SAAF,IAA  
e,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,QAAF,  
IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,eAAF,IAAqB,a;MACrB,EAAE,QAAF,IAAc,M;MACd,EAA  
E,SAAF,IAAe,O;MACf,EAAE,UAAF,IAAgB,Q;MACHB,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;M  
ACf,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,YAAF,IAAkB,U;MACIB,EA  
AE,gBAAF,IAAsB,c;MACtB,EAAE,eAAF,IAAqB,a;MACrB,EAAE,iBAAF,IAAuB,e;MACvB,EAAE,oBAAF,IA  
A0B,kB;MAC1B,EAAE,eAAF,IAAqB,a;MACrB,EAAE,gBAAF,IAAsB,c;MACtB,EAAE,oBAAF,IAA0B,kB;MA  
C1B,EAAE,MAAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IA  
AkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MACHB,OAAO,C;K;wGAgDX,kQ;MAEoC,uB;QAAA,UAAoB,K;MAA  
O,wB;QAAA,WAAqB,K;MAAO,sB;QAAA,SAAmB,K;MAAO,uB;QAAA,UAAoB,K;MAAO,gC;QAAA,mBAA  
6B,K;MAAO,gC;QAAA,mBAA6B,K;MAAO,0B;QAAA,aAAuB,K;MAAO,8B;QAAA,iBAA2B,K;MAAO,6B;QA  
AA,gBAA0B,K;MAAO,+B;QAAA,kBAA4B,K;MAAO,kC;QAAA,qBAA+B,K;MAAO,6B;QAAA,gBAA0B,K;M  
AAO,8B;QAAA,iBAA2B,K;MAAO,kC;QAAA,qBAA+B,K;MAAO,oB;QAAA,OAAgB,I;MAAM,sB;QAAA,SA  
Ae,C;MAAG,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MAC71B,QAA  
Q,E;MACR,EAAE,SAAF,IAAe,O;MACf,EAAE,UAAF,IAAgB,Q;MACHB,EAAE,QAAF,IAAc,M;MACd,EAAE,  
SAAF,IAAe,O;MACf,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,YAAF,IAA  
kB,U;MACIB,EAAE,gBAAF,IAAsB,c;MACtB,EAAE,eAAF,IAAqB,a;MACrB,EAAE,iBAAF,IAAuB,e;MACvB,  
EAAE,oBAAF,IAA0B,kB;MAC1B,EAAE,eAAF,IAAqB,a;MACrB,EAAE,gBAAF,IAAsB,c;MACtB,EAAE,oBA  
AF,IAA0B,kB;MAC1B,EAAE,MAAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf  
,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MACHB,OAAO,C;K;kGAsCX,iX;MAEiC,sB;QAAA,S  
AAkB,G;MAAK,sB;QAAA,SAAkB,G;MAAK,sB;QAAA,SAAkB,G;MAAK,yB;QAAA,YAAkB,C;MAAG,uB;Q  
AAA,UAAgB,C;MAAG,uB;QAAA,UAAgB,C;MAAG,uB;QAAA,UAAgB,C;MAAG,uB;QAAA,UAAgB,C;MAA  
G,sB;QAAA,SAAiB,C;MAAG,uB;QAAA,UAAkB,C;MAAG,6B;QAAA,gBAA8B,I;MAAM,sB;QAAA,SAAkB,I;  
MAAM,uB;QAAA,UAAoB,K;MAAO,wB;QAAA,WAAqB,K;MAAO,sB;QAAA,SAAmB,K;MAAO,uB;QAAA,U

AAoB,K;MAAO,gC;QAAA,mBAA6B,K;MAAO,gC;QAAA,mBAA6B,K;MAAO,0B;QAAA,aAAuB,K;MAAO,8B;QAAA,iBAA2B,K;MAAO,6B;QAAA,gBAA0B,K;MAAO,+B;QAAA,kBAA4B,K;MAAO,kC;QAAA,qBAA+B,K;MAAO,6B;QAAA,gBAA0B,K;MAAO,8B;QAAA,iBAA2B,K;MAAO,kC;QAAA,qBAA+B,K;MAAO,oB;QAAA,OAAgB,I;MAAM,sB;QAAA,SA Ae,C;MAAG,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACr2B,QAAQ,E;MACR,EAAE,QAAF,IAAc,M;MACd,EAAE,QAAF,IAAc,M;MACd,EAAE,QAAF,IAAc,M;MACd,EAAE,WAAf,IAAiB,S;MACjB,EAAE,SAAF,IAAe,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,eAAF,IAAqB,a;MACrB,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,UAAf,IAAgB,Q;MACHB,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,YAAF,IAAkB,U;MACIB,EAAE,gBAAF,IAAsB,c;MACtB,EAAE,eAAF,IAAqB,a;MACrB,EAAE,iBAAF,IAAuB,e;MACvB,EAAE,oBAAF,IAA0B,kB;MAC1B,EAAE,eAAF,IAAqB,a;MACrB,EAAE,gBAAF,IAAsB,c;MACtB,EAAE,oBAAF,IAA0B,kB;MAC1B,EAAE,MAAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAf,IAAgB,Q;MACHB,OAAO,C;K;kGA2BX,0E;MAEiC,oB;QAAA,OAAgB,E;MAAI,2B;QAAA,cAAwB,K;MAAO,oB;QAAA,OAAgB,I;MAAM,sB;QAAA,SA Ae,C;MAAG,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACtM,QAAQ,E;MACR,EAAE,MAAF,IAAY,I;MACZ,EAAE,aAAF,IAAmB,W;MACnB,EAAE,MAAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAf,IAAgB,Q;MACHB,OAAO,C;K;wGAmDX,4S;MAEoC,mB;QAAA,MAAe,E;MAAI,oB;QAAA,OAAgB,E;MAAI,wB;QAAA,WAAiB,C;MAAG,sB;QAAA,SAAmB,K;MAAO,2B;QAAA,cAAwB,K;MAAO,uB;QAAA,UAAoB,K;MAAO,wB;QAAA,WAAqB,K;MAAO,sB;QAAA,SAAmB,K;MAAO,uB;QAAA,UAAoB,K;MAAO,gC;QAAA,mBAA6B,K;MAAO,gC;QAAA,mBAA6B,K;MAAO,0B;QAAA,aAAuB,K;MAAO,8B;QAAA,iBAA2B,K;MAAO,6B;QAAA,gBAA0B,K;MAAO,+B;QAAA,kBAA4B,K;MAAO,kC;QAAA,qBAA+B,K;MAAO,6B;QAAA,gBAA0B,K;MAAO,8B;QAAA,iBAA2B,K;MAAO,kC;QAAA,qBAA+B,K;MAAO,oB;QAAA,OAAgB,I;MAAM,sB;QAAA,SA Ae,C;MAAG,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACjtB,QAAQ,E;MACR,EAAE,KAAF,IAAW,G;MACX,EAAE,MAAF,IAAY,I;MACZ,EAAE,UAAf,IAAgB,Q;MACHB,EAAE,QAAF,IAAc,M;MACd,EAAE,aAAF,IAAmB,W;MACnB,EAAE,SAAF,IAAe,O;MACf,EAAE,UAAf,IAAgB,Q;MACHB,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,YAAF,IAAkB,U;MACIB,EAAE,gBAAF,IAAsB,c;MACtB,EAAE,eAAF,IAAqB,a;MACrB,EAAE,iBAAF,IAAuB,e;MACvB,EAAE,oBAAF,IAA0B,kB;MAC1B,EAAE,eAAF,IAAqB,a;MACrB,EAAE,gBAAF,IAAsB,c;MACtB,EAAE,oBAAF,IAA0B,kB;MAC1B,EAAE,MAAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAf,IAAgB,Q;MACHB,OAAO,C;K;8GAuBX,6D;MAEuC,oB;QAAA,OAAgB,E;MAAI,oB;QAAA,OAAgB,I;MAAM,sB;QAAA,SA Ae,C;MAAG,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MAC7K,QAAQ,E;MACR,EAAE,MAAF,IAAY,I;MACZ,EAAE,MAAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAf,IAAgB,Q;MACHB,OAAO,C;K;wECnbX,4B;MAEyE,iBAAY,KAAZ,C;K;wEAEzE,2B;MAEgG,iBAAY,IAAZ,C;K;wEAwBhG,oC;MAE+F,UAA Y,KAAZ,IAAqB,M;K;wEAmFpH,2B;MAEqE,iBAAY,IAAZ,C;K;wEAErE,kC;MAE2E,UAA Y,IAAZ,IAAoB,K;K;wEAssC/F,4B;MAEyE,iBAAY,KAAZ,C;K;wEA0BzE,4B;MAEyE,iBAAY,KAAZ,C;K;wEAsBzE,4B;MAEuE,iB AAY,KAAZ,C;K;wEAyBvE,4B;MAE6E,iBAAY,KAAZ,C;K;2FA4C7E,gD;MAEiC,qB;QAAA,QAAiD,I;MAAM,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACIK,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAf,IAAgB,Q;MACHB,OAAO,C;K;uEA+UX,4B;MAEuE,iBAAY,KAAZ,C;K;wEAEvE,2B;MAE6F,iBAAY,IAAZ,C;K;wEAqN 7F,4B;MAEyE,iBAAY,KAAZ,C;K;wEAEzE,oC;MAE2F,UAA Y,KAAZ,IAAqB,M;K;+FAuehH,wD;MAEmC,6B;QAAA,gBAA8B,I;MAAM,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACjJ,QAAQ,E;MACR,EAAE,eAAF,IAAqB,a;MACrB,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAf,IAAgB,Q;MACHB,OAAO,C;K;uGAuIX,mB;MAEuC,uB;QAAA,UAAoB,K;MACvD,QAAQ,E;MACR,EAAE,SAAF,IAAe,O;MACf,OAAO,C;K;+HAyCX,iB;MAEmD,qB;QAAA,QAAkB,I;MACjE,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,OAAO,C;K;+FA0MX,sE;MAEmC,oB;QAAA,OAAgB,I;MAAM,wB;QAAA,

WA0+G4B,S;OA1+GwB,kB;QAAA,KAAc,E;MAAI,wB;QAAA,WAAoB,I;MAAM,sB;QAAA,SAAkB,S;MAAW,uB;QAAA,UAAoB,I;MAAM,qB;QAAA,QAAiB,I;MAAM,oB;QAAA,OAAgB,I;MACnP,QAAQ,E;MACR,EAAE,MAAF,IAAY,I;MACZ,EAAE,UAAF,IAAgB,Q;MACHb,EAAE,IAAF,IAAU,E;MACV,EAAE,UAAF,IAAgB,Q;MACHb,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,OAAF,IAAa,K;MACb,EAAE,MAAF,IAAY,I;MACZ,OAAO,C;K;qIAgDX,iB;MAEsD,qB;QAAA,QAakB,I;MACpE,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,OAAO,C;K;+GAKBX,qB;MAE2C,yB;QAAA,YAAmB,S;MAC1D,QAAQ,E;MACR,EAAE,SAAF,IAAe,S;MACf,OAAO,C;K;wEAKCX,4B;MAEqF,iBAAY,KAAZ,C;K;yFAgCrF,4V;MAEgC,4B;QAAA,eAA8B,I;MAAM,uB;QAAA,UAAgB,C;MAAG,uB;QAAA,UAAgB,C;MAAG,uB;QAAA,UAAgB,C;MAAG,uB;QAAA,UAAgB,C;MAAG,sB;QAAA,SAAiB,C;MAAG,uB;QAAA,UAAkB,C;MAAG,6B;QAAA,gBAA8B,I;MAAM,sB;QAAA,SAAkB,I;MAAM,uB;QAAA,UAAoB,K;MAAO,wB;QAAA,WAAqB,K;MAAO,sB;QAAA,SAAmB,K;MAAO,uB;QAAA,UAAoB,K;MAAO,gC;QAAA,mBAA6B,K;MAAO,gC;QAAA,mBAA6B,K;MAAO,0B;QAAA,aAAuB,K;MAAO,8B;QAAA,iBAA2B,K;MAAO,6B;QAAA,gBAA0B,K;MAAO,+B;QAAA,kBAA4B,K;MAAO,kC;QAAA,qBAA+B,K;MAAO,6B;QAAA,gBAA0B,K;MAAO,8B;QAAA,iBAA2B,K;MAAO,kC;QAAA,qBAA+B,K;MAAO,oB;QAAA,OAAgB,I;MAAM,sB;QAAA,SAAe,C;MAAG,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MAC9yB,QAAQ,E;MACR,EAAE,cAAF,IAAoB,Y;MACpB,EAAE,SAAF,IAAe,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,eAAF,IAAqB,a;MACrB,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,UAAF,IAAgB,Q;MACHb,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,YAAF,IAAkB,U;MACIB,EAAE,gBAAF,IAAsB,c;MACtB,EAAE,eAAF,IAAqB,a;MACrB,EAAE,iBAAF,IAAuB,e;MACvB,EAAE,oBAAF,IAA0B,kB;MACIB,EAAE,eAAF,IAAqB,a;MACrB,EAAE,gBAAF,IAAsB,c;MACtB,EAAE,oBAAF,IAA0B,kB;MACIB,EAAE,MAAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;wEAWEX,2B;MAE+D,iBAAY,IAAZ,C;K;giA2D/D,gD;MAEOC,qB;QAAA,QAAC,I;MAAM,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACII,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;qGA2BX,yD;MAEsC,sB;QAAA,SAAkB,E;MAAI,sB;QAAA,SAAkB,E;MAAI,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MAC5J,QAAQ,E;MACR,EAAE,QAAF,IAAc,M;MACd,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;6GAuBX,oD;MAEOC,yB;QAAA,YAAsB,K;MAAO,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACjJ,QAAQ,E;MACR,EAAE,WAAF,IAAiB,S;MACjB,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;2FAoFX,kF;MAEiC,uB;QAAA,UAAmB,E;MAAI,wB;QAAA,WAAoB,E;MAAI,sB;QAAA,SAAe,C;MAAG,qB;QAAA,QAAC,C;MAAG,qB;QAAA,QAAC,I;MAAM,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACjN,QAAQ,E;MACR,EAAE,SAAF,IAAe,O;MACf,EAAE,UAAF,IAAgB,Q;MACHb,EAAE,QAAF,IAAc,M;MACd,EAAE,OAAF,IAAa,K;MACb,EAAE,OAAF,IAAa,K;MACb,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;iHAyBX,0D;MAEqE,sB;QAAA,SAAe,S;MAAW,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACzK,QAAQ,E;MACR,EAAE,SAAF,IAAe,O;MACf,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;wEAmXX,4B;MAEkE,iBAAY,KAAZ,C;K;wEAEIE,2B;MAEOE,iBAAY,IAAZ,C;K;wEAUPE,4B;MAEsE,iBAAY,KAAZ,C;K;wEAETe,2B;MAEW,e,iBAAY,IAAZ,C;K;wEAaxE,4B;MAE+D,iBAAY,KAAZ,C;K;wEAE/D,2B;MAEiE,iBAAY,IAAZ,C;K;mGA0CjE,8G;MAEqC,gC;QAAA,mBAooF8C,M;OApOFe,gC;QAAA,mBAmpFT,S;OAnpFyE,oC;QAAA,uBA8pFjE,S;OA9pF6I,2B;QAAA,cAAoB,S;MAAW,4B;QAAA,eAAqB,S;MAAW,6B;QAAA,gBAyqFIO,K;OAxqFvE,QAAQ,E;MACR,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,sBAAF,IAA4B,oB;MAC5B,EAAE,aAAF,IAAmB,W;MACnB,EAAE,cAAF,IAAoB,Y;MACpB,EAAE,eAAF,IAAqB,a;MACrB,OAAO,C;K;+FAwCX,mF;MAEmC,oB;QAAA,OAAa,I;MAAM,sB;QAAA,SAAkB,E;MAAI,2B;QAAA,cAAuB,E;MAAI,sB;QAAA,SAAyC,I;MAAM,qB;QAAA,QAA6B,E;MAAW,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACxQ,QAAQ,E;MACR,EAAE,MAAF,IAAY,I;MA

CZ,EAAE,QAAF,IAAc,M;MACd,EAAE,aAAF,IAAmB,W;MACnB,EAAE,QAAF,IAAc,M;MACd,EAAE,OAAF,IAAa,K;MACb,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACiB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;6FA4BX,2B;MAEkC,+B;QAAA,kBAA4B,K;MAC1D,QAAQ,E;MACR,EAAE,iBAAF,IAAuB,e;MACvB,OAAO,C;K;2FA2DX,iE;MAEiC,wB;QAAA,WAAqB,K;MAAO,oB;QAAA,OAAe,C;MAAG,sB;QAAA,SAAKB,E;MAAI,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MAC/K,QAAQ,E;MACR,EAAE,UAAF,IAAgB,Q;MACHb,EAAE,MAAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACiB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;yFA8FX,6B;MAEgC,oB;QAAA,OA+7E6C,S;OA/7EL,2B;QAAA,cCl2He,M;ODm2HnF,QAAQ,E;MACR,EAAE,MAAF,IAAY,I;MACZ,EAAE,aAAF,IAAmB,W;MACnB,OAAO,C;K;wEAoDX,0B;MAE+D,iBAAY,GAZ,C;K;wEAE/D,iC;MAEqE,UAAAY,GAZ,IAAmB,K;K;+FAoDxF,oF;MAEmC,mB;QAAA,MAAe,I;MAAM,wB;QAAA,WAAoB,I;MAAM,wB;QAAA,WAAoB,I;MAAM,mB;QAAA,MAAe,E;MAAI,2B;QAAA,cAAwB,I;MAAM,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACvO,QAAQ,E;MACR,EAAE,KAAF,IAAW,G;MACX,EAAE,UAAF,IAAgB,Q;MACHb,EAAE,UAAF,IAAgB,Q;MACHb,EAAE,KAAF,IAAW,G;MACX,EAAE,aAAF,IAAmB,W;MACnB,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACiB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;iFAwNX,yC;MAE4B,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACiG,QAAQ,E;MACR,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACiB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;6FAwBX,iD;MAEkC,sB;QAAA,SA Ae,I;MAAM,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACjI,QAAQ,E;MACR,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACiB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;uGASX,mB;MAEuC,uB;QAAA,UAAoB,K;MACvD,QAAQ,E;MACR,EAAE,SAAF,IAAe,O;MACf,OAAO,C;K;6GAYX,kC;MAE0C,uB;QAAA,UAAoB,K;MAAO,oB;QAAA,OAAiB,K;MAAO,uB;QAAA,UAAoB,K;MAC7G,QAAQ,E;MACR,EAAE,SAAF,IAAe,O;MACf,EAAE,MAAF,IAAY,I;MACZ,EAAE,SAAF,IAAe,O;MACf,OAAO,C;K;wEAkEX,4B;MAE6D,iBAAY,KAAZ,C;K;wEAU7D,4B;MAEsE,iBAAY,KAAZ,C;K;wEAeT,2B;MAEwE,iBAAY,IAAZ,C;K;uGAsCxE,oH;MAEuC,yB;QAAA,YAAsB,K;MAAO,0B;QAAA,aAAuB,S;MAAW,6B;QAAA,gBAA0B,S;MAAW,uB;QAAA,UAAoB,K;MAAO,iC;QAAA,oBAA8B,S;MAAW,qC;QAAA,wBAaK,C,S;MAAW,+B;QAAA,kBAaK,C,S;MAC1R,QAAQ,E;MACR,EAAE,WAAF,IAAiB,S;MACjB,EAAE,YAAF,IAAkB,U;MACiB,EAAE,eAAF,IAAqB,a;MACrB,EAAE,SAAF,IAAe,O;MACf,EAAE,mBAAF,IAAYB,iB;MACzB,EAAE,uBAAF,IAA6B,qB;MAC7B,EAAE,iBAAF,IAAuB,e;MACvB,OAAO,C;K;mGAgFX,oB;MAEqC,wB;QAAA,WAAqB,K;MACtD,QAAQ,E;MACR,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;wEA+MX,2B;MAEiE,iBAAY,IAAZ,C;K;2GakCjE,c;MAEyC,kB;QAAA,KAAgB,S;MACrD,QAAQ,E;MACR,EAAE,IAAF,IAAU,E;MACV,OAAO,C;K;2FAuMX,gB;MAGI,QAAQ,E;MACR,EAAE,MAAF,IAAY,I;MACZ,OAAO,C;K;wEAgBX,4B;MAEiE,iBAAY,KAAZ,C;K;wEAejE,oC;MAE4E,iBAAY,aAAZ,C;K;wEAuT5E,4B;MAEmE,iBAAY,KAAZ,C;K;uFA2CnE,sB;MAE+B,iB;QAAA,IAAa,G;MAAK,iB;QAAA,IAAa,G;MAAK,iB;QAAA,IAAa,G;MAAK,iB;QAAA,IAAa,G;MAC9F,QAAQ,E;MACR,EAAE,GAAF,IAAS,C;MACT,EAAE,GAAF,IAAS,C;MACT,EAAE,GAAF,IAAS,C;MACT,EAAE,GAAF,IAAS,C;MACT,OAAO,C;K;qFA0CX,+B;MAE8B,iB;QAAA,IAAa,G;MAAK,iB;QAAA,IAAa,G;MAAK,qB;QAAA,QAAiB,G;MAAK,sB;QAAA,SAAKB,G;MACtG,QAAQ,E;MACR,EAAE,GAAF,IAAS,C;MACT,EAAE,GAAF,IAAS,C;MACT,EAAE,OAAF,IAAa,K;MACb,EAAE,QAAF,IAAc,M;MACd,OAAO,C;K;wEAOX,4B;MAEmE,iBAAY,KAAZ,C;K;yFAiHnE,oB;MAEgC,wB;QAAA,WAY2B+C,M;OAx2B3E,QAAQ,E;MACR,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;6FAeX,+B;MAEkC,oB;QAAA,OAAGB,S;MAAW,mB;QAAA,MAAe,S;MAAW,wB;QAAA,Waq1BR,M;OAp1B3E,QAAQ,E;MACR,EAAE,MAAF,IAAY,I;MACZ,EAAE,KAAF,IAAW,G;MACX,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;6GAwCX,yD;MAE0C,qB;QAAA,QAAiB,E;MAAI,uB;QAAA,UAAoB,K;MAAO,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACpK,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,SAAF,IAAe,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACiB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;yGAiCX,mC;MAEwC,qB;QAAA,QA2wByD,Q;OA3wBK,sB;QAAA,SA2wBL,Q;OA3wBoE,wB;QAAA,WA4vBtF,M;OA3vB3E,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,QAAF,IAAc,M;MACd,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;2FAYX,2B;MAEiC,mB;QAAA,MAuwB0C,Q;OAvwBJ,0B;QAAA,aAAsB,S;MACzF,QAAQ,E;MACR,EAAE,KAAF,IAAW,G;MACX,EAAE,YAAF,IAAkB,U;MACiB,OAAO,C;K;+GAYX,0B;MAE2C,uB;QAAA,UaqvBgC,Q;OArvBU,q

B;QAAA,QAqvBV,Q;OApvBvE,QAAQ,E;MACR,EAAE,SAAF,IAAe,O;MACf,EAAE,OAAF,IAAa,K;MACb,OA  
AO,C;K;wEAgCX,4B;MAE+D,iBAAY,KAZ,C;K;qFAyaY,qB;MAAQ,OAAU,S;K;6FAEd,qB;MAAQ,OAAc,a;  
K;uFAEzB,qB;MAAQ,OAAW,U;K;iFASxB,qB;MAAQ,OAAG,E;K;iFAEX,qB;MAAQ,OAAQ,O;K;uFAEb,qB;M  
AAQ,OAAW,U;K;uFAS3B,qB;MAAQ,OAAW,U;K;mFAErB,qB;MAAQ,OAAS,Q;K;qFAEhB,qB;MAAQ,OAAU  
,S;K;yFAShB,qB;MAAQ,OAA Y,W;K;uFAErB,qB;MAAQ,OAAW,U;K;+FAEf,qB;MAAQ,OAAe,c;K;uFAE3B,q  
B;MAAQ,OAAW,U;K;uFAEnB,qB;MAAQ,OAAW,U;K;mFASrB,qB;MAAQ,OAAS,Q;K;iFAEIB,qB;MAAQ,OA  
AQ,O;K;6EAEIB,qB;MAAQ,OAAM,K;K;uFAET,qB;MAAQ,OAAW,U;K;qFASIB,qB;MAAQ,OAAU,S;K;qFAE  
IB,qB;MAAQ,OAAU,S;K;6EASR,qB;MAAQ,OAAM,K;K;mFAEX,qB;MAAQ,OAAS,Q;K;+EAEnB,qB;MAAQ,O  
AAO,M;K;+EAS/B,qB;MAAQ,OAAO,M;K;iFAEd,qB;MAAQ,OAAQ,O;K;mFAEf,qB;MAAQ,OAAS,Q;K;mFAS  
hB,qB;MAAQ,OAAQ,O;K;iFAEhB,qB;MAAQ,OAAQ,O;K;iFAEhB,qB;MAAQ,OAAQ,O;K;mFASd,qB;MAAQ,  
OAAQ,O;K;+EAEIB,qB;MAAQ,OAAM,K;K;+EAEb,qB;MAAQ,OAAO,M;K;iFAEd,qB;MAAQ,OAAQ,O;K;mF  
AEf,qB;MAAQ,OAAS,Q;K;6EASd,qB;MAAQ,OAAM,K;K;qFAEV,qB;MAAQ,OAAU,S;K;mFAEnB,qB;MAAQ,  
OAAS,Q;K;2FAEb,qB;MAAQ,OAAa,Y;K;6FAEpB,qB;MAAQ,OAAc,a;K;mFAE3B,qB;MAAQ,OAAS,Q;K;6EA  
S1B,qB;MAAQ,OAAM,K;K;6EAEEd,qB;MAAQ,OAAM,K;K;qFAEV,qB;MAAQ,OAAU,S;K;+EASjB,qB;MAAQ,  
OAAO,M;K;mFAEb,qB;MAAQ,OAAS,Q;K;+EASrB,qB;MAAQ,OAAO,M;K;iFAEd,qB;MAAQ,OAAQ,O;K;iFA  
SjB,qB;MAAQ,OAAO,M;K;6FAER,qB;MAAQ,OAAc,a;K;qFAE1B,qB;MAAQ,OAAU,S;K;iFASb,qB;MAAQ,O  
AAO,M;K;uFAEZ,qB;MAAQ,OAAU,S;K;yFAS9B,qB;MAAQ,OAA Y,W;K;+EAE1B,qB;MAAQ,OAAM,K;K;qF  
AEX,qB;MAAQ,OAAS,Q;K;iFAEnB,qB;MAAQ,OAAO,M;K;+EASrB,qB;MAAQ,OAAO,M;K;6FAER,qB;MAA  
Q,OAAc,a;K;qFAS1B,qB;MAAQ,OAAU,S;K;mFAEnB,qB;MAAQ,OAAS,Q;K;+EASX,qB;MAAQ,OAAO,M;K;  
mFAEb,qB;MAAQ,OAAS,Q;K;iFASnB,qB;MAAQ,OAAO,M;K;qFAEZ,qB;MAAQ,OAAU,S;K;mFAEnB,qB;M  
AAQ,OAAS,Q;K;kFASJ,qB;MAAQ,OAAQ,O;K;oFAEf,qB;MAAQ,OAAS,Q;K;8EAEpB,qB;MAAQ,OAAM,K;K  
;oFAEV,qB;MAAQ,OAAU,S;K;mFASzC,qB;MAAQ,OAAS,Q;K;mFAEjB,qB;MAAQ,OAAS,Q;K;qFAEhB,qB;M  
AAQ,OAAU,S;K;qFAEIB,qB;MAAQ,OAAU,S;K;wIEx+M7E,wM;MAEiD,qB;QAAA,QAakB,I;MAAM,sB;QAA  
A,SAAmB,I;MAAM,2B;QAAA,cAAwB,I;MAAM,yB;QAAA,YAAaB,I;MAAM,0B;QAAA,aAAuB,I;MAAM,0B;  
QAAA,aAAuB,I;MAAM,sB;QAAA,SAAmB,I;MAAM,0B;QAAA,aAAuB,I;MAAM,0B;QAAA,aAAuB,I;MAAM,  
gC;QAAA,mBAA6B,I;MAAM,+B;QAAA,kBAA4B,I;MAAM,gC;QAAA,mBAA6B,I;MAAM,uB;QAAA,UAAoB  
,I;MAAM,4B;QAAA,eAAyB,I;MAAM,wB;QAAA,WAAqB,I;MAAM,uB;QAAA,UAAoB,I;MACrf,QAAQ,E;MA  
CR,EAAE,OAAF,IAAa,K;MACb,EAAE,QAAF,IAAc,M;MACd,EAAE,aAAF,IAAmB,W;MACnB,EAAE,WAAF,  
IAAiB,S;MACjB,EAAE,YAAF,IAAkB,U;MACiB,EAAE,YAAF,IAAkB,U;MACiB,EAAE,QAAF,IAAc,M;MACd  
,EAAE,YAAF,IAAkB,U;MACiB,EAAE,YAAF,IAAkB,U;MACiB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,iB  
AAF,IAAuB,e;MACvB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,SAAF,IAAe,O;MACf,EAAE,cAAF,IAAoB,Y;  
MACpB,EAAE,UAAF,IAAgB,Q;MACHB,EAAE,SAAF,IAAe,O;MACf,OAAO,C;K;wHAsDX,wM;MAEyC,qB;Q  
AAA,QAAqB,S;MAAW,sB;QAAA,SAAsB,S;MAAW,2B;QAAA,cAA4B,S;MAAW,yB;QAAA,YAA0B,S;MAA  
W,0B;QAAA,aAA6B,S;MAAW,0B;QAAA,aAA6B,S;MAAW,sB;QAAA,SAAuB,S;MAAW,0B;QAAA,aAA0B,S;  
MAAW,0B;QAAA,aAA0B,S;MAAW,gC;QAAA,mBAAoC,S;MAAW,+B;QAAA,kBAAmC,S;MAAW,gC;QAAA  
,mBAAoC,S;MAAW,uB;QAAA,UAAwB,S;MAAW,4B;QAAA,eAA4B,S;MAAW,wB;QAAA,WAAoB,S;MAAW  
,uB;QAAA,UAAmB,S;MACtnB,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,QAAF,IAAc,M;MACd,E  
AAE,aAAF,IAAmB,W;MACnB,EAAE,WAAF,IAAiB,S;MACjB,EAAE,YAAF,IAAkB,U;MACiB,EAAE,YAAF,I  
AAkB,U;MACiB,EAAE,QAAF,IAAc,M;MACd,EAAE,YAAF,IAAkB,U;MACiB,EAAE,YAAF,IAAkB,U;MACiB  
,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,iBAAF,IAAuB,e;MACvB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,  
SAAF,IAAe,O;MACf,EAAE,cAAF,IAAoB,Y;MACpB,EAAE,UAAF,IAAgB,Q;MACHB,EAAE,SAAF,IAAe,O;M  
ACf,OAAO,C;K;sHAYX,kN;MAEwC,wB;QAAA,WAA4C,S;MAAW,qB;QAAA,QAAiB,S;MAAW,sB;QAAA,S  
AAkB,S;MAAW,2B;QAAA,cAAuB,S;MAAW,yB;QAAA,YAAqB,S;MAAW,0B;QAAA,aAsB,S;MAAW,0B;Q  
AAA,aAsB,S;MAAW,sB;QAAA,SAakB,S;MAAW,0B;QAAA,aAsB,S;MAAW,0B;QAAA,aAsB,S;MAAW,  
gC;QAAA,mBAA4B,S;MAAW,+B;QAAA,kBAA2B,S;MAAW,gC;QAAA,mBAA4B,S;MAAW,uB;QAAA,UAA  
mB,S;MAAW,4B;QAAA,eAAwB,S;MAAW,wB;QAAA,WAAoB,S;MAAW,uB;QAAA,UAAmB,S;MAC9IB,QA  
AQ,E;MACR,EAAE,UAAF,IAAgB,Q;MACHB,EAAE,OAAF,IAAa,K;MACb,EAAE,QAAF,IAAc,M;MACd,EAA  
E,aAAF,IAAmB,W;MACnB,EAAE,WAAF,IAAiB,S;MACjB,EAAE,YAAF,IAAkB,U;MACiB,EAAE,YAAF,IAA

kB,U;MACIB,EAAE,QAAF,IAAc,M;MACd,EAAE,YAAF,IAAkB,U;MACIB,EAAE,YAAF,IAAkB,U;MACIB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,iBAAF,IAAuB,e;MACvB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,SAAF,IAAe,O;MACf,EAAE,cAAF,IAAoB,Y;MACpB,EAAE,UAAF,IAAgB,Q;MAChB,EAAE,SAAF,IAAe,O;MACf,OAAO,C;K;0HAsDX,wM;MAE0C,qB;QAAA,QAAiB,S;MAAW,sB;QAAA,SAAkB,S;MAAW,2B;QAAA,cAAuB,S;MAAW,yB;QAAA,YAAqB,S;MAAW,0B;QAAA,aAAsB,S;MAAW,0B;QAAA,aAAsB,S;MAAW,sB;QAAA,SAAkB,S;MAAW,0B;QAAA,aAAsB,S;MAAW,0B;QAAA,aAAsB,S;MAAW,gC;QAAA,mBAA4B,S;MAAW,+B;QAAA,kBAA2B,S;MAAW,gC;QAAA,mBAA4B,S;MAAW,uB;QAAA,UAAmB,S;MAAW,4B;QAAA,eAAwB,S;MAAW,wB;QAAA,WAAoB,S;MAAW,uB;QAAA,UAAmB,S;MACziB,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,QAAF,IAAc,M;MACd,EAAE,aAAF,IAAmB,W;MACnB,EAAE,WAAF,IAAiB,S;MACjB,EAAE,YAAF,IAAkB,U;MACIB,EAAE,YAAF,IAAkB,U;MACIB,EAAE,QAAF,IAAc,M;MACd,EAAE,YAAF,IAAkB,U;MACIB,EAAE,YAAF,IAAkB,U;MACIB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,iBAAF,IAAuB,e;MACvB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,SAAF,IAAe,O;MACf,EAAE,cAAF,IAAoB,Y;MACpB,EAAE,UAAF,IAAgB,Q;MAChB,EAAE,SAAF,IAAe,O;MACf,OAAO,C;K;gHAYDX,wM;MAEqC,qB;QAAA,QAAc,S;MAAW,sB;QAAA,SA Ae,S;MAAW,2B;QAAA,cAAuB,S;MAAW,yB;QAAA,YAAqB,S;MAAW,0B;QAAA,aAAsB,S;MAAW,0B;QAAA,aAAsB,S;MAAW,sB;QAAA,SAAkB,S;MAAW,0B;QAAA,aAAmB,S;MAAW,0B;QAAA,aAAmB,S;MAAW,gC;QAAA,mBAA6B,S;MAAW,+B;QAAA,kBAA4B,S;MAAW,gC;QAAA,mBAA6B,S;MAAW,uB;QAAA,UAAmB,S;MAAW,4B;QAAA,eAAqB,S;MAAW,wB;QAAA,WAAoB,S;MAAW,uB;QAAA,UAAmB,S;MACxhB,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,QAAF,IAAc,M;MACd,EAAE,aAAF,IAAmB,W;MACnB,EAAE,WAAF,IAAiB,S;MACjB,EAAE,YAAF,IAAkB,U;MACIB,EAAE,YAAF,IAAkB,U;MACIB,EAAE,QAAF,IAAc,M;MACd,EAAE,YAAF,IAAkB,U;MACIB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,iBAAF,IAAuB,e;MACvB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,SAAF,IAAe,O;MACf,EAAE,cAAF,IAAoB,Y;MACpB,EAAE,UAAF,IAAgB,Q;MAChB,EAAE,SAAF,IAAe,O;MACf,OAAO,C;K;8HAqBX,gD;MAEsE,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACHJ,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MAChB,OAAO,C;K;sIAoBX,gD;MAEgD,qB;QAAA,QAAiB,I;MAAM,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACjJ,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MAChB,OAAO,C;K;wHAWCX,wB;MAEyC,qB;QAAA,QAAiB,K;MAAO,qB;QAAA,QAAiB,K;MAC9E,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,OAAF,IAAa,K;MACb,OAAO,C;K;kGAYBX,oB;MAE8B,mB;QAAA,MAAe,S;MAAW,mB;QAAA,MAAe,S;MACnE,QAAQ,E;MACR,EAAE,KAAF,IAAW,G;MACX,EAAE,KAAF,IAAW,G;MACX,OAAO,C;K;oHAYX,kC;MAEuC,qB;QAAA,QAAiB,S;MAAW,qB;QAAA,QAAiB,S;MAAW,mB;QAAA,MAAe,S;MAAW,mB;QAAA,MAAe,S;MACpI,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,OAAF,IAAa,K;MACb,EAAE,KAAF,IAAW,G;MACX,EAAE,KAAF,IAAW,G;MACX,OAAO,C;K;gGAYX,oB;MAE6B,mB;QAAA,MAAY,S;MAAW,mB;QAAA,MAAY,S;MAC5D,QAAQ,E;MACR,EAAE,KAAF,IAAW,G;MACX,EAAE,KAAF,IAAW,G;MACX,OAAO,C;K;kHAYX,kC;MAEsC,qB;QAAA,QAAc,S;MAAW,qB;QAAA,QAAc,S;MAAW,mB;QAAA,MAAY,S;MAAW,mB;QAAA,MAAY,S;MACvH,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,OAAF,IAAa,K;MACb,EAAE,KAAF,IAAW,G;MACX,EAAE,KAAF,IAAW,G;MACX,OAAO,C;K;gIAeX,wB;MAE6C,qB;QAAA,QAAkB,S;MAAW,qB;QAAA,QAAkB,S;MACxF,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,OAAF,IAAa,K;MACb,OAAO,C;K;oIAeX,wB;MAE+C,qB;QAAA,QAAiB,S;MAAW,qB;QAAA,QAAiB,S;MACxF,QAAQ,E;MACR,EAAE,OAAF,IAAa,K;MACb,EAAE,OAAF,IAAa,K;MACb,OAAO,C;K;4FAKX,Y;MAGI,QAAQ,E;MACR,OAAO,C;K;oFAKX,Y;MAGI,QAAQ,E;MACR,OAAO,C;K;8FAKX,Y;MAGI,QAAQ,E;MACR,OAAO,C;K;kGASX,oB;MAE8B,wB;QAAA,WAAkC,S;MAC5D,QAAQ,E;MACR,EAAE,UAAF,IAAgB,Q;MAChB,OAAO,C;K;4FAUmE,qB;MAAQ,OAAO,M;K;8FAEd,qB;MAAQ,OAAQ,O;K;4FASrB,qB;MAAQ,OAAO,M;K;0GAER,qB;MAAQ,OAAc,a;K;8FAE7B,qB;MAAQ,OAAO,M;K;gGAEd,qB;MAAQ,OAAQ,O;K;8FASjB,qB;MAAQ,OAAO,M;K;gHAEL,qB;MAAQ,OAAiB,gB;K;wGASrC,qB;MAAQ,OAAa,Y;K;0GAEPB,qB;MAAQ,OAAc,a;K;wGAEvB,qB;MAAQ,OAAa,Y;K;oFCroB7F,4B;MAE6E,iBAAY,KAAZ,C;K;iGASnB,qB;MAAQ,OAAS,Q;K;6FAEnB,qB;MAAQ,OAAO,M;K;+FAEd,qB;MAAQ,OAAQ,O;K;iGASF,qB;MAAQ,OAAU,S;K;+FAEnB,qB;MAAQ,OAAS,Q;K;mGAS3B,qB;MAAQ,OAAW,U;K;mGAEnB,qB;MAAQ,OAAW,U;K;6GC1D/E,mb;MAEmC,

yB;QAAA,YAAkB,C;MAAG,qB;QAAA,QAAiB,G;MAAK,sB;QAAA,SAAkB,G;MAAK,wB;QAAA,WAAmB,G;  
MAAI,kC;QAAA,qBAA6B,G;MAAI,qB;QAAA,QAAc,C;MAAG,qB;QAAA,QAAc,C;MAAG,qB;QAAA,QAAc,  
C;MAAG,2B;QAAA,cAAuB,E;MAAI,yB;QAAA,YAAsB,K;MAAO,uB;QAAA,UAAgB,C;MAAG,uB;QAAA,UA  
AgB,C;MAAG,uB;QAAA,UAAgB,C;MAAG,uB;QAAA,UAAgB,C;MAAG,sB;QAAA,SAAiB,C;MAAG,uB;QAA  
A,UAAkB,C;MAAG,6B;QAAA,gBAA8B,I;MAAM,sB;QAAA,SAAkB,I;MAAM,uB;QAAA,UAAoB,K;MAAO,w  
B;QAAA,WAAqB,K;MAAO,sB;QAAA,SAAmB,K;MAAO,uB;QAAA,UAAoB,K;MAAO,gC;QAAA,mBAA6B,K  
;MAAO,gC;QAAA,mBAA6B,K;MAAO,0B;QAAA,aAAuB,K;MAAO,8B;QAAA,iBAA2B,K;MAAO,6B;QAAA,g  
BAA0B,K;MAAO,+B;QAAA,kBAA4B,K;MAAO,kC;QAAA,qBAA+B,K;MAAO,6B;QAAA,gBAA0B,K;MAAO,  
8B;QAAA,iBAA2B,K;MAAO,kC;QAAA,qBAA+B,K;MAAO,oB;QAAA,OAAgB,I;MAAM,sB;QAAA,SA Ae,C;  
MAAG,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACl/B,QAAQ,E;M  
ACR,EAAE,WAAF,IAAiB,S;MACjB,EAAE,OAAF,IAAa,K;MACb,EAAE,QAAF,IAAc,M;MACd,EAAE,UAAF,  
IAAgB,Q;MACHB,EAAE,oBAAF,IAA0B,kB;MAC1B,EAAE,OAAF,IAAa,K;MACb,EAAE,OAAF,IAAa,K;MAC  
b,EAAE,OAAF,IAAa,K;MACb,EAAE,aAAF,IAAmB,W;MACnB,EAAE,WAAF,IAAiB,S;MACjB,EAAE,SAAF,I  
AAe,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,SAAF,IAAe,O;MACf,EAAE,QA  
AF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,eAAF,IAAqB,a;MACrB,EAAE,QAAF,IAAc,M;MACd,E  
AAE,SAAF,IAAe,O;MACf,EAAE,UAAF,IAAgB,Q;MACHB,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O  
;MACf,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,YAAF,IAAkB,U;MACIB,  
EAAE,gBAAF,IAAsB,c;MACtB,EAAE,eAAF,IAAqB,a;MACrB,EAAE,iBAAF,IAAuB,e;MACvB,EAAE,oBAAF,  
IAA0B,kB;MAC1B,EAAE,eAAF,IAAqB,a;MACrB,EAAE,gBAAF,IAAsB,c;MACtB,EAAE,oBAAF,IAA0B,kB;  
MAC1B,EAAE,MAAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,  
IAAkB,U;MACIB,EAAE,UAAF,IAAgB,Q;MACHB,OAAO,C;K;6GC1BX,0C;MAEwC,oB;QAAA,OAAiB,I;MAA  
M,sB;QAAA,SAAmB,K;MAAO,uB;QAAA,UAAoB,K;MAAO,uB;QAAA,UAAoB,K;MACpI,QAAQ,E;MACR,E  
AAE,MAAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,SAAF,IAAe,O;M  
ACf,OAAO,C;K;4EAmIX,4B;MAEKe,iBAAY,KAAZ,C;K;4EAEIE,qC;MAE2E,UAA Y,KAAZ,IAAqB,O;K;4EAI  
BhG,4B;MAEuE,iBAAY,KAAZ,C;K;4EAEvE,qC;MAE+E,UAA Y,KAAZ,IAAqB,O;K;4EAI BpG,4B;MAEuE,iBA  
AY,KAAZ,C;K;4EAEvE,qC;MAE+E,UAA Y,KAAZ,IAAqB,O;K;4EAI GpG,4B;MAEoE,iBAAY,KAAZ,C;K;2EA  
EpE,qC;MAE4E,UAA Y,KAAZ,IAAqB,O;K;4EAKcJG,4B;MAE6E,iBAAY,KAAZ,C;K;4EAE7E,qC;MAEqF,UAA  
Y,KAAZ,IAAqB,O;K;4EAgP1G,4B;MAEqE,iBAAY,KAAZ,C;K;4EAErE,qC;MAE6E,UAA Y,KAAZ,IAAqB,O;K  
;uFJ57BIG,+H;MAE8B,sB;QAAA,SAAkB,S;MAAW,uB;QAAA,UAAmB,S;MAAW,oB;QAAA,OAAgB,S;MAA  
W,wB;QAAA,WAAoB,S;MAAW,8B;QAAA,iBAA0B,S;MAAW,oB;QAAA,OAAqB,S;MAAW,2B;QAAA,cAAm  
C,S;MAAW,qB;QAAA,QAAuB,S;MAAW,wB;QAAA,WAA6B,S;MAAW,yB;QAAA,YAAqB,S;MAAW,yB;QA  
AA,YAAsB,S;MAAW,wB;QAAA,WAAe,S;MAC5Z,QAAQ,E;MACR,EAAE,QAAF,IAAc,M;MACd,EAAE,SA  
AF,IAAe,O;MACf,EAAE,MAAF,IAAY,I;MACZ,EAAE,UAAF,IAAgB,Q;MACHB,EAAE,gBAAF,IAAsB,c;MACt  
B,EAAE,MAAF,IAAY,I;MACZ,EAAE,aAAF,IAAmB,W;MACnB,EAAE,OAAF,IAAa,K;MACb,EAAE,UAAF,IA  
AgB,Q;MACHB,EAAE,WAAF,IAAiB,S;MACjB,EAAE,WAAF,IAAiB,S;MACjB,EAAE,QAAF,IAAc,Q;MACd,O  
AAO,C;K;yFA0CX,uC;MAE+B,sB;QAAA,SAAiB,G;MAAK,0B;QAAA,aAAsB,I;MAAM,uB;QAAA,UAAmB,S;  
MACHG,QAAQ,E;MACR,EAAE,QAAF,IAAc,M;MACd,EAAE,YAAF,IAAkB,U;MACIB,EAAE,SAAF,IAAe,O;  
MACf,OAAO,C;K;qFAUgD,qB;MAAQ,OAAQ,E;K;mFAEX,qB;MAAQ,OAAQ,O;K;iFAEjB,qB;MAAQ,OAAO,  
M;K;mFAEd,qB;MAAQ,OAAQ,O;K;qFAEf,qB;MAAQ,OAAS,Q;K;mFAEIB,qB;MAAQ,OAAQ,O;K;mFAEhB,q  
B;MAAQ,OAAQ,O;K;mFAEhB,qB;MAAQ,OAAQ,O;K;qFASF,qB;MAAQ,OAAQ,E;K;yFAER,qB;MAAQ,OAA  
W,U;K;mFAEtB,qB;MAAQ,OAAQ,O;K;mFAEjB,qB;MAAQ,OAAO,M;K;qFAEd,qB;MAAQ,OAAQ,O;K;yFAEb  
B,qB;MAAQ,OAAW,U;K;mFAEtB,qB;MAAQ,OAAQ,O;K;qFAEf,qB;MAAQ,OAAS,Q;K;qFAEjB,qB;MAAQ,OA  
AS,Q;K;uFAEjB,qB;MAAQ,OAAS,Q;K;mGAEV,qB;MAAQ,OAAgB,e;K;iGAEzB,qB;MAAQ,OAAe,c;K;qFAE9  
B,qB;MAAQ,OAAQ,O;K;qFAEf,qB;MAAQ,OAAS,Q;K;iFAEnB,qB;MAAQ,OAAO,M;K;yFASzB,qB;MAAQ,O  
AAW,U;K;+FAEhB,qB;MAAQ,OAAc,a;K;uFAE1B,qB;MAAQ,OAAU,S;K;iFAErB,qB;MAAQ,OAAO,M;K;iFA  
SD,qB;MAAQ,OAAO,M;K;iGAER,qB;MAAQ,OAAc,a;K;uFAE1B,qB;MAAQ,OAAU,S;K;yFAS9B,qB;MAAQ,  
OAAU,S;K;yFAEjB,qB;MAAQ,OAAW,U;K;qFAErB,qB;MAAQ,OAAS,Q;K;yFAEf,qB;MAAQ,OAAW,U;K;+F  
AEhB,qB;MAAQ,OAAc,a;K;qGAEnB,qB;MAAQ,OAAiB,gB;K;qFAS3B,qB;MAAQ,OAAS,Q;K;mFAEIB,qB;M

AAQ,OAAQ,O;K;uFAEf,qB;MAAQ,OAAS,Q;K;mFASxB,qB;MAAQ,OAAQ,O;K;mFAEjB,qB;MAAQ,OAAO,  
M;K;yFAEZ,qB;MAAQ,OAAU,S;K;qFAEpB,qB;MAAQ,OAAQ,O;K;qFAEf,qB;MAAQ,OAAS,Q;K;qGAET,qB;  
MAAQ,OAAiB,gB;K;+FKnR/F,gB;MAEkC,oB;QAAA,OAAgB,E;MAC9C,QAAQ,E;MACR,EAAE,MAAF,IAA  
Y,I;MACZ,OAAO,C;K;+FAiBX,8B;MAEkC,4B;QAAA,eAAqB,S;MAAW,oB;QAAA,OAAgB,E;MAC9E,QAAQ,  
E;MACR,EAAE,cAAF,IAAoB,Y;MACpB,EAAE,MAAF,IAAY,I;MACZ,OAAO,C;K;0EAUX,4B;MAE6D,iBAA  
Y,KA AZ,C;K;+GC6B7D,sJ;MAEsC,mB;QAAA,MA4GuD,M;OA5GG,oB;QAAA,OAAgB,E;MAAI,oB;QAAA,O  
AAgB,E;MAAI,mB;QAAA,MAAe,E;MAAI,qB;QAAA,QAAiB,S;MAAW,oB;QAAA,OAAgB,S;MAAW,qB;QA  
AA,QAAiB,S;MAAW,qB;QAAA,QAAiB,S;MAAW,uB;QAAA,UAAmB,S;MAAW,yB;QAAA,YAAqB,S;MAA  
W,wB;QAAA,WAAqB,K;MAAO,sB;QAAA,SAAmB,K;MAAO,wB;QAAA,WAAqB,K;MAAO,kC;QAAA,qBAA  
+B,K;MAAO,sB;QAAA,SAAmB,K;MAAO,oB;QAAA,OAAa,I;MAAM,uB;QAAA,UAAc,E;MAC/gB,QAAQ,E;  
MACR,EAAE,KAAF,IAAW,G;MACX,EAAE,MAAF,IAAY,I;MACZ,EAAE,MAAF,IAAY,I;MACZ,EAAE,KAA  
F,IAAW,G;MACX,EAAE,OAAF,IAAa,K;MACb,EAAE,MAAF,IAAY,I;MACZ,EAAE,OAAF,IAAa,K;MACb,EA  
AE,OAAF,IAAa,K;MACb,EAAE,SAAF,IAAe,O;MACf,EAAE,WAAF,IAAiB,S;MACjB,EAAE,UAAF,IAAgB,Q;  
MACHb,EAAE,QAAF,IAAc,M;MACd,EAAE,UAAF,IAAgB,Q;MACHb,EAAE,oBAAF,IAA0B,kB;MACiB,EAA  
E,QAAF,IAAc,M;MACd,EAAE,MAAF,IAAY,I;MACZ,EAAE,SAAF,IAAe,O;MACf,OAAO,C;K;6GAWX,+B;M  
AEsE,oB;QAAA,OAAgB,S;MACiF,QAAQ,E;MACR,EAAE,QAAF,IAAc,M;MACd,EAAE,OAAF,IAAa,K;MAC  
b,EAAE,MAAF,IAAY,I;MACZ,OAAO,C;K;qHASX,e;MAEyC,mB;QAAA,MAAe,E;MACpD,QAAQ,E;MACR,E  
AAE,KAAF,IAAW,G;MACX,OAAO,C;K;mHAyBX,+D;MAEqE,sB;QAAA,SAAkB,E;MAAI,uB;QAAA,UAAoB  
,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACrK,QAAQ,E;MACR,EAAE,cAAF,IAAoB,Y  
;MACpB,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACiB,EAAE,UA  
AF,IAAgB,Q;MACHb,OAAO,C;K;iGAUwE,qB;MAAQ,OAAU,S;K;6FAEnB,qB;MAAQ,OAAS,Q;K;+FAEhB,q  
B;MAAQ,OAAU,S;K;2FASvB,qB;MAAQ,OAAO,M;K;yFAEhB,qB;MAAQ,OAAM,K;K;yFAEd,qB;MAAQ,OA  
AM,K;K;yGCrJ3F,uB;MAEsC,qB;QAAA,QAAiB,S;MAAW,oB;QAAA,ORy9MW,S;OQx9MzE,QAAQ,E;MACR  
,EAAE,OAAF,IAAa,K;MACb,EAAE,MAAF,IAAY,I;MACZ,OAAO,C;K;6HAuCX,mF;MAEgD,oB;QAAA,OAA  
a,S;MAAW,sB;QAAA,SAAkB,S;MAAW,2B;QAAA,cAAuB,S;MAAW,sB;QAAA,SAA2C,S;MAAW,qB;QAAA,  
QAA6B,S;MAAW,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MAC/S,Q  
AAQ,E;MACR,EAAE,MAAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,aAAF,IAAmB,W;MACnB,E  
AAE,QAAF,IAAc,M;MACd,EAAE,OAAF,IAAa,K;MACb,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;  
MACiB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;uGA2DX,qC;MAEqC,mC;QAAA,sBAAGC,K;MAAO,oB;  
QAAA,OA4UD,Q;OA3UvE,QAAQ,E;MACR,EAAE,qBAAF,IAA2B,mB;MAC3B,EAAE,MAAF,IAAY,I;MACZ,  
OAAO,C;K;yGAmBX,yC;MAEsC,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAq  
B,K;MACHh,QAAQ,E;MACR,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACiB,EAAE,UAAF,IAAg  
B,Q;MACHb,OAAO,C;K;yGAsBX,2B;MAGI,QAAQ,E;MACR,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe  
,O;MACf,OAAO,C;K;+FA8BX,sE;MAEoD,wB;QAAA,WAAoB,I;MAAM,wB;QAAA,WAAqB,K;MAAO,uB;QA  
AA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACpL,QAAQ,E;MACR,EAAE,SA  
AF,IAAe,O;MACf,EAAE,UAAF,IAAgB,Q;MACHb,EAAE,UAAF,IAAgB,Q;MACHb,EAAE,SAAF,IAAe,O;MA  
Cf,EAAE,YAAF,IAAkB,U;MACiB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;6GAuBX,0D;MAE2D,sB;QAA  
A,SAAkB,M;MAAQ,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MAC/J,  
QAAQ,E;MACR,EAAE,SAAF,IAAe,O;MACf,EAAE,QAAF,IAAc,M;MACd,EAAE,SAAF,IAAe,O;MACf,EAAE,  
YAAF,IAAkB,U;MACiB,EAAE,UAAF,IAAgB,Q;MACHb,OAAO,C;K;2GAaX,qC;MAE4D,sB;QAAA,SAAkB,S;  
MAAW,uB;QAAA,UAA0B,S;MAC/G,QAAQ,E;MACR,EAAE,UAAF,IAAgB,Q;MACHb,EAAE,QAAF,IAAc,M;  
MACd,EAAE,SAAF,IAAe,O;MACf,OAAO,C;K;uHAuCX,mF;MAE6C,oB;QAAA,OAAa,S;MAAW,sB;QAAA,S  
AAkB,S;MAAW,2B;QAAA,cAAuB,S;MAAW,sB;QAAA,SAAmD,S;MAAW,qB;QAAA,QAA6B,S;MAAW,uB;Q  
AAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA,WAAqB,K;MACpT,QAAQ,E;MACR,EAAE,M  
AAF,IAAY,I;MACZ,EAAE,QAAF,IAAc,M;MACd,EAAE,aAAF,IAAmB,W;MACnB,EAAE,QAAF,IAAc,M;MA  
Cd,EAAE,OAAF,IAAa,K;MACb,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACiB,EAAE,UAAF,IA  
AgB,Q;MACHb,OAAO,C;K;qGA+BX,6D;MAEoC,4B;QAAA,eAAyB,K;MAAO,4B;QAAA,eAAyB,K;MAAO,0B  
;QAAA,aAAuB,K;MAAO,yB;QAAA,YAAqB,S;MACnJ,QAAQ,E;MACR,EAAE,cAAF,IAAoB,Y;MACpB,EAA

E,cAAF,IAAoB,Y;MACpB,EAAE,YAAF,IAAkB,U;MACiB,EAAE,WAAF,IAAiB,S;MACjB,OAAO,C;K;yGAKB X,4C;MAEsC,oB;QAAA,OAAgB,S;MAAW,uB;QAAA,UAAoB,S;MAAW,wB;QAAA,WAAaB,S;MAAW,uB;QA AA,UAA8B,S;MAC3J,QAAQ,E;MACR,EAAE,MAAF,IAAY,I;MACZ,EAAE,SAAF,IAAe,O;MACf,EAAE,UAA F,IAAgB,Q;MAChB,EAAE,SAAF,IAAe,O;MACf,OAAO,C;K;+FAkCmE,qB;MAAQ,OAAa,Y;K;6FAEtB,qB;MA AQ,OAAy,W;K;+FAEnB,qB;MAAQ,OAAa,Y;K;6FAEtB,qB;MAAQ,OAAy,W;K;6FAEpB,qB;MAAQ,OAAy, W;K;6FAStC,qB;MAAQ,OAAy,W;K;6FAEpB,qB;MAAQ,OAAy,W;K;uFAEvB,qB;MAAQ,OAAS,Q;K;qFAEn B,qB;MAAQ,OAAO,M;K;uFASX,qB;MAAQ,OAAS,Q;K;yFAEjB,qB;MAAQ,OAAS,Q;K;qGAEX,qB;MAAQ,O AAE,c;K;iFAEhC,qB;MAAQ,OAAM,K;K;iGCharE,0E;MAEOC,gC;QAAA,mBAA6B,K;MAAO,sB;QAAA,SAAk B,C;MAAG,qB;QAAA,QAAiB,C;MAAG,uB;QAAA,UAAoB,K;MAAO,0B;QAAA,aAAuB,K;MAAO,wB;QAAA ,WAAqB,K;MAC3L,QAAQ,E;MACR,EAAE,kBAAF,IAAwB,gB;MACxB,EAAE,QAAF,IAAc,M;MACd,EAAE, OAAF,IAAa,K;MACb,EAAE,SAAF,IAAe,O;MACf,EAAE,YAAF,IAAkB,U;MACiB,EAAE,UAAF,IAAgB,Q;MA ChB,OAAO,C;K;mFAU8E,qB;MAAQ,OAAG,E;K;+FAEL,qB;MAAQ,OAAC,a;K;iFAE7B,qB;MAAQ,OAAO,M; K;yFAEX,qB;MAAQ,OAAW,U;K;+EAEvB,qB;MAAQ,OAAO,M;K;+EAEf,qB;MAAQ,OAAO,M;K;oErIjIvG,yB ;MAAA,kF;MAAA,0B;MAAA,uB;QAAI,IAAI,OAAO,CAAP,IAA8B,OAAO,KAAzC,C;UACI,MAAM,8BAAYB, wBAAqB,IAA9C,C;SAEV,OAAy,OAAL,IAAK,C;O;KAhBhB,C;0EAwCiC,qB;MAAQ,OAAA,SAAK,I;K;IsIpB V,6B;MAAC,qB;QAAA,8C;MAAA,kB;K;IACjC,2C;MAAA,e;MAAA,iB;MAAA,uB;K;IAAA,yC;MAAA,4C;O; MAKI,0E;MAEA,sE;K;;IAFA,kD;MAAA,+B;MAAA,0C;K;;IAEA,gD;MAAA,+B;MAAA,wC;K;;IAPJ,qC;MAA A,yF;K;;IAAA,0C;MAAA,a;AAAA,S;UAAA,+C;aAAA,O;UAAA,6C;gBAAA,8D;;K;;IAyBmC,sC;MACnC,8B;K ;IAMqC,sC;MACrC,8B;K;;IC1DJ,iC;K;;ICMA,4B;K;;IA6BA,gD;K;;IC5BA,qC;K;;IA4BA,+B;K;;ICRqC,uC;MA CjC,uB;QAAA,UAAaB,E;MACtB,qB;QAAA,+C;MADA,sB;MACA,kB;K;IAEA,4C;MAAA,e;MAAA,iB;MAAA, uB;K;IAAA,0C;MAAA,6C;O;MAKI,4E;MAGA,wE;K;;IAHA,mD;MAAA,gC;MAAA,2C;K;;IAGA,iD;MAAA,gC ;MAAA,yC;K;;IARJ,sC;MAAA,2F;K;;IAAA,2C;MAAA,a;AAAA,S;UAAA,gD;aAAA,O;UAAA,8C;gBAAA,+D;; K;;IAyByB,4B;MACzB,8B;K;;IC/C4C,8B;K;kDAI5C,mB;MAA6D,c;;QpJ2rD7C,Q;QADhB,IAAI,mCAAsB,cAA 1B,C;UAAqC,aAAO,K;UAAp,e;SACrB,sB;QAaHb,OAAgB,cAAhB,C;UAAgB,2B;UAAM,IoJ3rD6C,OpJ2rD/B, SoJ3rD+B,UpJ2rD7C,C;YAAwB,aAAO,I;YAAP,e;;QAC9C,aAAO,K;;MoJ5rDsD,iB;K;uDAE7D,oB;MACa,c;;Qp JmqDG,Q;QADhB,IAAI,coJlqDA,QpJkqDA,iBoJlqDA,QpJkqDsB,UAA1B,C;UAAqC,aAAO,I;UAAp,e;SACrB,O oJnqDZ,QpJmqDY,W;QAaHb,OAAgB,cAAhB,C;UAAgB,yB;UAAM,IAAI,CoJnqDP,oBpJmqDkB,OoJnqDIB,Cp JmqDG,C;YAAyB,aAAO,K;YAAP,e;;QAC/C,aAAO,I;;MoJpqDH,iB;K;2CAEJ,Y;MAAkC,qBAAQ,C;K;IAEqB,q E;MAAA,qB;QAC3D,OAAI,OAAO,uBAAX,GAAiB,mBAAjB,GAA6C,SAAH,EAAG,C;O;K;4CADjD,Y;MAAk C,4BAAa,IAAb,EAAMb,GAAAnB,EAAwB,GAAxB,kBAA6B,wCAA7B,C;K;2CAIIC,Y;MAI4C,uBAAgB,IAAhB, C;K;mDAE5C,iB;MAI4D,yBAAgB,IAAhB,EAAsB,KAAtB,C;K;;IC/BhE,8B;MAAA,e;MAAA,iB;MAAA,uB;K;I AAA,4B;MAAA,+B;O;MACI,4C;MACA,kD;MACA,0C;MACA,8C;K;;IAHA,mC;MAAA,kB;MAAA,2B;K;;IAC A,sC;MAAA,kB;MAAA,8B;K;;IACA,kC;MAAA,kB;MAAA,0B;K;;IACA,oC;MAAA,kB;MAAA,4B;K;;IAJJ,wB; MAAA,sH;K;;IAAA,6B;MAAA,a;AAAA,O;UAAA,gC;aAAA,U;UAAA,mC;aAAA,M;UAAA,+B;aAAA,Q;UAAA ,iC;gBAAA,6D;;K;;IAOA,4B;MAKI,mD;MACA,2BAA4B,I;K;yCAE5B,Y;MAEiB,IAAN,I;M3JUX,IAAI,E2JXQ, mD3JWR,CAAJ,C;QACI,cAda,qB;QAEb,MAAM,gCAAYB,OAAQ,WAAjC,C;O2JZC,QAAM,oBAAN,M;aACH, M;UAAc,Y;UAAAd,K;aACA,O;UAAe,W;UAAf,K;gBACQ,wC;UAHL,K;;MAAP,W;K;sCAOJ,Y;MAIW,Q;MAHP ,IAAI,CAAC,cAAL,C;QAAgB,MAAM,6B;MACtB,mD;MAEA,OAAO,2F;K;4DAGX,Y;MACI,iD;MACA,kB;M ACA,OAAO,kD;K;+CAeX,iB;MAII,2BAAY,K;MACZ,gD;K;sCAGJ,Y;MAII,+C;K;;ICjDkC,wB;MAoFtC,oC;MA pFgE,6B;K;sCAIhE,Y;MAAuC,0C;K;2CAEvC,mB;MAAwD,uB;;QtJkU3C,Q;QADb,YAAy,C;QACC,sB;QAAb, OAAa,cAAb,C;UAAa,sB;UACT,IsJnUmE,OtJmUrD,IsJnUqD,UtJmUnE,C;YACI,sBAAO,K;YAAP,wB;WACJ,qB ;QAEJ,sBAAO,E;;MsJvUiD,0B;K;+CAExD,mB;MAA4D,sB;;QtJ2V5D,eAAoB,0BAAa,SAAb,C;QACpB,OAAO ,QAAS,cAAhB,C;UACI,IsJ7VsE,OtJ6VxD,QAAS,WsJ7V+C,UtJ6VtE,C;YACI,qBAAO,QAAS,Y;YAAhB,uB;;Q AGR,qBAAO,E;;MsJWqD,yB;K;0CAE5D,Y;MAA+C,+CAAiB,CAAjB,C;K;kDAE/C,iB;MAAyD,+CAAiB,KA AjB,C;K;6CAEzD,8B;MAA8D,gCAAQ,IAAR,EAAC,SAAd,EAAyB,OAAzB,C;K;IAEIC,wD;MAAgF,uB;MAA/E ,kB;MAAMc,4B;MAC5D,eAAyB,C;MAGrB,+DAAkB,gBAAIB,EAA6B,OAA7B,EAAsC,WAAK,KAA3C,C;MA CA,eAAa,UAAU,gBAAV,I;K;iDAGjB,iB;MACI,+DAAkB,KAAIB,EAAyB,YAAzB,C;MAEA,OAAO,wBAAK,m BAAY,KAAZ,IAAL,C;K;4FAGY,Y;MAAQ,mB;K;;oCAGnC,iB;MAMI,IAAI,UAAU,IAAd,C;QAAoB,OAAO,I;

MAC3B,IAAI,2BAAJ,C;QAAuB,OAAO,K;MAE9B,OAAO,2DAAc,IAAd,EAAoB,KAApB,C;K;sCAGX,Y;MAG +B,oEAAgB,IAAhB,C;K;IAE/B,2C;MAAA,oB;MACI,eACsB,C;K;kDAEtB,Y;MAAkC,sBAAQ,gB;K;+CAE1C,Y ;MAEe,gB;MADX,IAAI,CAAC,cAAL,C;QAAgB,MAAM,6B;MACX,iE;MAAX,OAAO,+B;K;;IAO0B,sD;MAHz C,oB;MAGwD,iD;MAGhD,gEAAmB,KAAmB,EAA0B,WAAkB,KAA5C,C;MACA,eAAa,K;K;0DAGjB,Y;MAAs C,sBAAQ,C;K;wDAE9C,Y;MAAgC,mB;K;uDAEhC,Y;MACI,IAAI,CAAC,kBAAL,C;QAAoB,MAAM,6B;MAC 1B,OAAO,yBAAI,mCAAJ,EAAI,YAAJ,E;K;4DAGX,Y;MAAoC,sBAAQ,CAAR,I;K;;IAGxC,kC;MAAA,sC;K;iE ACI,uB;MACI,IAAI,QAAQ,CAAR,IAAa,SAAS,IAA1B,C;QACI,MAAM,8BAA0B,YAAS,KAAT,gBAAuB,IAAj D,C;Q;kEAIId,uB;MACI,IAAI,QAAQ,CAAR,IAAa,QAAQ,IAAzB,C;QACI,MAAM,8BAA0B,YAAS,KAAT,gBA AuB,IAAjD,C;Q;iEAIId,oC;MACI,IAAI,YAA Y,CAAZ,IAAiB,UAAU,IAA/B,C;QACI,MAAM,8BAA0B,gBAAa,S AAb,mBAaKc,OAAIC,gBAaKd,IAA5E,C;OAEV,IAAI,YAA Y,OAAhB,C;QACI,MAAM,gCAAYB,gBAAa,SAAb,oBAAmC,OAA5D,C;Q;kEAIId,sC;MACI,IAAI,aAAa,CAAb,IAAkB,WAAW,IAAjC,C;QACI,MAAM,8BAA0B,i BAAc,UAAAd,oBAAqC,QAArC,gBAAsD,IAAhF,C;OAEV,IAAI,aAAa,QAAjB,C;QACI,MAAM,gCAAYB,iBAAc, UAAAd,qBAAsC,QAA/D,C;Q;+DAId,a;MAEc,UACsB,M;MAFhC,iBA Ae,C;MACL,mB;MAAV,OAAU,cAAV,C; QAAU,mB;QACN,aAAW,MAAK,UAAAL,SAAiB,6DAAiB,CAAIC,K;;MAEf,OAAO,U;K;6DAGX,oB;MAIiB,Q; MAHb,IAAI,CAAE,KAAF,KAAU,KAAM,KAApB,C;QAA0B,OAAO,K;MAEjC,oBAAoB,KAAM,W;MACb,mB ;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,gBAAGB,aAAc,O;QAC9B,IAAI,cAAQ,SAAR,CAAJ,C;UACI,OAAO,K; ;MAGf,OAAO,I;K;;IAjDf,8C;MAAA,6C;QAAA,4B;OAAA,sC;K;;ICnFwC,uB;MAyHxC,mC;MAzCA,uBAC6B,I ;MAmC7B,yBACsC,I;K;8CAnHtC,e;MACI,OAAO,6BAAc,GAAd,S;K;gDAGX,iB;MAAwE,gBAAR,Y;MAAQ,c; ;QvJkrDxD,Q;QADhB,IAAI,wCAAsB,mBAA1B,C;UAAqC,aAAO,K;UAAP,e;SACrB,2B;QAAhB,OAAgB,cAAh B,C;UAAgB,yB;UAAM,IuJlrDwD,OvJkrD1C,OuJlrD6C,MAAH,QvJkrDxD,C;YAAwB,aAAO,I;YAAP,e;;QAC9 C,aAAO,K;;MuJnrDyD,iB;K;kDAEhE,iB;MAEI,IAAI,gCAAJ,C;QAA+B,OAAO,K;MACtC,UAAU,KAAM,I;MA ChB,YAA Y,KAAM,M;MpKiNO,Q;MoKhNzB,epKgN4C,CAAnB,mDAAmB,YoKhNzB,GpKgNyB,C;MoK9M5C ,IAAI,eAAS,QAAT,CAAJ,C;QACI,OAAO,K;OAIp,6B;MAAA,W;QpK0NqB,U;QoK1ND,UpK0NoB,CAAnB,uD AAmB,oBoK1NP,GpK0NO,C;OoK1N5C,W;QACI,OAAO,K;OAGX,OAAO,I;K;mCAIX,iB;MAMI,IAAI,UAAU,I AAd,C;QAAoB,OAAO,I;MAC3B,IAAI,0BAAJ,C;QAAyB,OAAO,K;MACHC,IAAI,cAAQ,KAAM,KAAIB,C;QA AwB,OAAO,K;MAEV,gBAAd,KAAM,Q;MAAQ,c;;QvJ6nDT,Q;QADhB,IAAI,wCAAsB,mBAA1B,C;UAAqC,a AAO,I;UAAP,e;SACrB,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UAAM,IAAI,CuJ7nDK,2BvJ6nDM,OuJ7nDN ,CvJ6nDT,C;YAAyB,aAAO,K;YAAP,e;;QAC/C,aAAO,I;;MuJ9nDH,iB;K;sCAGJ,e;MAAwC,Q;MAAA,4CAAc, GAAd,8B;K;qCAGxC,Y;MAK+B,OAAQ,SAAR,YAAQ,C;K;oCAEvC,Y;MAAkC,qBAAQ,C;K;mFACnB,Y;MA AQ,OAAA,YAAQ,K;K;IAWnB,0E;MAAA,wC;MAAS,sB;K;8EAcB,mB;MAAsD,+CAAY,OAAZ,C;K;IAI3C,sG; MAAA,kD;K;8FACH,Y;MAAkC,OAAA,0BAAc,U;K;2FACHD,Y;MAAyB,OAAA,0BAAc,OAAO,I;K;;wEAJtD, Y;MACI,oBAAoB,6BAAQ,W;MAC5B,+F;K;sHAMmB,Y;MAAQ,OAAA,qBAAiB,K;K;;mFAB5D,Y;MACI,IAAI ,4BAAJ,C;QACI,+E;OAcJ,OAAO,mC;K;IAOwD,uD;MAAA,qB;QAAE,2CAAS,EAAT,C;O;K;qCAAzE,Y;MAAk C,OAAQ,eAAR,YAAQ,EAAa,IAAb,EAAmB,GAAnB,EAAwB,GAAXB,kBAA6B,iCAA7B,C;K;+CAE1C,iB;MA AuD,+BAAS,KAAM,IAAf,IAAsB,GAAtB,GAA4B,wBAAS,KAAM,MAAf,C;K;+CAEnF,a;MAAwC,OAAI,MA AM,IAAV,GAAgB,YAAhB,GAAoC,SAAF,CAAE,C;K;IAWtD,4E;MAAA,wC;MAAS,6B;K;gFACf,mB;MAAsE, iDAAc,OAAAd,C;K;IAI3D,wG;MAAA,kD;K;gGACH,Y;MAAkC,OAAA,0BAAc,U;K;6FACHD,Y;MAAyB,OAAA ,0BAAc,OAAO,M;K;;0EAJtD,Y;MACI,oBAAoB,6BAAQ,W;MAC5B,iG;K;wHAMmB,Y;MAAQ,OAAA,qBAAi B,K;K;;qFAB5D,Y;MACI,IAAI,8BAAJ,C;QACI,mF;OAcJ,OAAO,qC;K;oDAMf,e;MAA8D,gBAAR,Y;MAAQ,sB ;;QvJjJ9C,Q;QAAA,2B;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UAAM,IuJjJsD,OvJjJxC,OuJjJ2C,IAAH,MvJjJtD, C;YAAwB,qBAAO,O;YAAP,uB;;QAC9C,qBAAO,I;;MuJlJ+C,yB;K;IAEtD,iC;MAAA,qC;K;4DAEI,a;MAAiE,g C;MAAX,OAAU,CAAC,kBAAN,CAAM,0DAAmB,CAApB,KAA4B,oBAAjC,CAAiC,8DAAqB,CAAjD,C;K;4D AChE,a;MAAyD,OAAU,SAAL,CAAO,IAAF,mBAAL,CAAY,MAAP,C;K;0DACnE,oB;MACI,IAAI,gCAAJ,C;Q AA+B,OAAO,K;MACtC,OAAO,OAAA,CAAE,IAAF,EAAS,KAAM,IAAf,KAA5B,OAAA,CAAE,MAAF,EAAW ,KAAM,MAAjB,C;K;;IANrC,6C;MAAA,4C;QAAA,2B;OAAA,qC;K;;IChIqC,uB;MAkBrC,mC;MAIB+D,6B;K; mCAE/D,iB;MAMI,IAAI,UAAU,IAAd,C;QAAoB,OAAO,I;MAC3B,IAAI,0BAAJ,C;QAA5B,OAAO,K;MAC7B, OAAO,sDAAU,IAAV,EAAgB,KAAhB,C;K;qCAGX,Y;MAG+B,qEAAkB,IAAiB,C;K;IAE/B,iC;MAAA,qC;K;gE ACI,a;MAEoB,Q;MADhB,iBA Ae,C;MACC,mB;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACC,U;QAAb,2BAAa

,yEAAuB,CAApC,K;;MAEJ,OAAO,U;K;wDAGX,oB;MACI,IAAI,CAAE,KAAF,KAAU,KAAM,KAAPB,C;QAA  
0B,OAAO,K;MACjC,OAAO,CtK40sG,qBsK5OxF,KtK40wF,C;K;;;IsKvPrH,6C;MAAA,4C;QAAA,2B;OAAA,q  
C;K;;;MCghBA,kC;MA9hBA,cAAwB,C;MACxB,yB;MAEA,sBAAyB,C;;kFAAZB,Y;MAAA,0B;K,OAAA,gB;M  
AAA,0B;K;4CA8BA,uB;MAOI,IAAI,cAAc,CAAlB,C;QAAqB,MAAM,6BAAsB,mBAAtB,C;MAC3B,IAAI,eAA  
e,kBAAY,OAA/B,C;QAAqC,M;MACrC,IAAI,uBAAgB,qDAAPB,C;QACI,qBAAc,gBAAyB,gBAAZ,WAAy,EA  
Ac,EAAd,CAAZB,O;QACd,M;OAGJ,kBAAkB,uDAAY,kBAAY,OAAxB,EAA8B,WAA9B,C;MACIB,oBAAa,W  
AAb,C;K;0CAGJ,uB;MAII,kBAAkB,gBAAMB,WAAAnB,O;M9J20BtB,U8J10BI,kB9J00BJ,E8J10ByB,W9J00BzB  
,E8J10BsC,C9J00BtC,E8J10ByC,W9J00BzC,E8J10B+C,kBAAY,O9J00B3D,C;MAAA,U8Jz0BI,kB9Jy0BJ,E8Jz0  
ByB,W9Jy0BzB,E8Jz0BsC,kBAAY,OAAZ,GAAmB,WAAAnB,I9Jy0BtC,E8Jz0B+D,C9Jy0B/D,E8Jz0BkE,W9Jy0  
BIE,C;M8Jx0BI,cAAO,C;MACP,qBAAc,W;K;yCAGIB,yB;MAGW,Q;MAAP,OAAO,2BAAY,aAAZ,4D;K;yCAG  
X,iB;MAA2C,OAAI,SAAS,kBAAY,OAAZB,GAA+B,QAAQ,kBAAY,OAApB,IAA/B,GAA6D,K;K;yCAEXG,iB;  
MAA2C,OAAI,QAAQ,CAAZ,GAAe,QAAQ,kBAAY,OAApB,IAAf,GAA6C,K;K;2CAEXF,iB;MACoD,0BAAY,c  
AAO,KAAP,IAAZ,C;K;yCAEPd,iB;MAA2C,OAAI,UAAqB,CAAZ,kBAAY,CAAzB,GAAoC,CAAPC,GAA2C,Q  
AAQ,CAAR,I;K;yCAEtF,iB;MAA2C,OAAI,UAAZ,CAAb,GAA4B,cAAZ,kBAAY,CAA5B,GAA2C,QAAQ,CAA  
R,I;K;mCAEtF,Y;MAAkC,qBAAQ,C;K;iCAE1C,Y;MAGwB,IAAI,cAAJ,C;QAAe,MAAM,2BAAuB,sBAAvB,C;;  
QAnBIC,Q;QAmBa,OAnBb,2BAmBkG,WAnBiG,4D;;K;uCAqBX,Y;MAG+B,Q;MAAA,IAAI,cAAJ,C;QAAA,O  
AAe,I;;QAxBnC,U;QAwBoB,OAxBpB,6BAwByD,WAxBzD,gE;;MAwBoB,W;K;gCAE/B,Y;MAGuB,IAAI,cAAJ  
,C;QAAe,MAAM,2BAAuB,sBAAvB,C;;QA7BjC,Q;QA6BY,OA7BZ,2BAQyC,mBAAY,cAqB0D,sBArB1D,IAA  
Z,CARzC,4D;;K;sCA+BX,Y;MAG8B,Q;MAAA,IAAI,cAAJ,C;QAAA,OAAe,I;;QAICIC,U;QAKcMB,OAlCnB,6B  
AQyC,mBAAY,cA0BiB,sBA1BjB,IAAZ,CARzC,gE;;MAkCmB,W;K;0CAE9B,mB;MAII,sBAAE,YAAO,CAAP,I  
AAf,C;MAEA,cAAO,mBAAY,WAAZ,C;MACP,mBAAY,WAAZ,IAAoB,O;MACpB,wBAAQ,CAAR,I;K;yCAGJ  
,mB;MAII,sBAAE,YAAO,CAAP,IAAf,C;MAEA,mBA7CgD,mBAAY,cA6CIC,SA7CkC,IAAZ,CA6ChD,IAAmC,  
O;MACnC,wBAAQ,CAAR,I;K;uCAGJ,Y;MAII,IAAI,cAAJ,C;QAAe,MAAM,2BAAuB,sBAAvB,C;MA7Dd,Q;M  
A+DP,cA/DO,2BA+DmB,WA/DnB,4D;MAGeP,mBAAY,WAAZ,IAAoB,I;MACpB,cAAO,mBAAY,WAAZ,C;M  
ACP,wBAAQ,CAAR,I;MACA,OAAO,O;K;6CAGX,Y;MAGqC,OAAI,cAAJ,GAAe,IAAf,GAAyB,kB;K;sCAE9D,  
Y;MAII,IAAI,cAAJ,C;QAAe,MAAM,2BAAuB,sBAAvB,C;MAErB,wBAzEgD,mBAAY,cAyEtB,sBAzEsB,IAAZ  
,C;MARzC,Q;MAkFP,cAlFO,2BAkFmB,iBAIfnB,4D;MAMFP,mBAAY,iBAAZ,IAAiC,I;MACjC,wBAAQ,CAA  
R,I;MACA,OAAO,O;K;4CAGX,Y;MAGoC,OAAI,cAAJ,GAAe,IAAf,GAAyB,iB;K;qCAE7D,mB;MAEI,mBAAQ  
,OAAr,C;MACA,OAAO,I;K;uCAGX,0B;MACI,oCAAA,4BAAMB,KAAAnB,EAA0B,SA1B,C;MAEb,IAAI,UAA  
S,SAAb,C;QACI,mBAAQ,OAAr,C;QACA,M;aACG,IAAI,UAAZ,CAAb,C;QACH,oBAAS,OAAT,C;QACA,M;O  
AGJ,sBAAE,YAAO,CAAP,IAAf,C;MA2BA,oBAjIgD,mBAAY,cAi1B,KAjI0B,IAAZ,C;MAMlhD,IAAI,QAAS,S  
AAD,GAAQ,CAAR,IAAE,CAA3B,C;QAEI,+BAA+B,mBAAY,aAAZ,C;QAC/B,sBAAsB,mBAAY,WAAZ,C;QA  
EtB,IAAI,4BAA4B,WAAhC,C;UACI,mBAAY,eAAZ,IAA+B,mBAAY,WAAZ,C;U9JgrB3C,U8J/qBY,kB9J+qBZ,  
E8J/qBiC,kB9J+qBjC,E8J/qB8C,W9J+qB9C,E8J/qBoD,cAAO,CAAP,I9J+qBpD,E8J/qB8D,2BAA2B,CAA3B,I9J  
+qB9D,C;;UAAA,U8J7qBY,kB9J6qBZ,E8J7qBiC,kB9J6qBjC,E8J7qB8C,cAAO,CAAP,I9J6qB9C,E8J7qBwD,W9  
J6qBxD,E8J7qB8D,kBAAY,O9J6qB1E,C;U8J5qBY,mBAAY,kBAAY,OAAZ,GAAmB,CAAnB,IAAZ,IAAoC,mB  
AAAY,CAAZ,C;U9J4qBhD,U8J3qBY,kB9J2qBZ,E8J3qBiC,kB9J2qBjC,E8J3qB8C,C9J2qB9C,E8J3qBiD,C9J2qBj  
D,E8J3qBoD,2BAA2B,CAA3B,I9J2qBpD,C;;Q8JxqBQ,mBAAY,wBAAZ,IAAwC,O;QACxC,cAAO,e;;QAGP,W  
ArJ4C,mBAAY,cAqJ/B,SArJ+B,IAAZ,C;QAUJ5C,IAAI,gBAAGB,IAAPB,C;U9JkqBR,U8JjqBY,kB9JjqBZ,E8Jjq  
BiC,kB9JjqBjC,E8JjqB8C,gBAAGB,CAAhB,I9JiqB9C,E8JjqBiE,a9JiqBjE,E8JjqBgF,I9JiqBhF,C;;UAAA,U8J/pB  
Y,kB9J+pBZ,E8J/pBiC,kB9J+pBjC,E8J/pB8C,C9J+pB9C,E8J/pBiD,C9J+pBjD,E8J/pBoD,I9J+pBpD,C;U8J9pBY,  
mBAAY,CAAZ,IAAiB,mBAAY,kBAAY,OAAZ,GAAmB,CAAnB,IAAZ,C;U9J8pB7B,U8J7pBY,kB9J6pBZ,E8J7  
pBiC,kB9J6pBjC,E8J7pB8C,gBAAGB,CAAhB,I9J6pB9C,E8J7pBiE,a9J6pBjE,E8J7pBgF,kBAAY,OAAZ,GAAm  
B,CAAnB,I9J6pBhF,C;;Q8J1pBQ,mBAAY,aAAZ,IAA6B,O;;MAEjC,wBAAQ,CAAR,I;K;oDAGJ,mC;MAGkD,U  
AIxB,M;MANtB,eAAe,QAAS,W;MAEsB,OAAZ,kBAAY,O;MAA9C,iBAAc,aAAd,wB;QACI,IAAI,CAAC,QAAS,  
UAAAd,C;UAAyB,K;QACzB,mBAAY,KAAZ,IAAqB,QAAS,O;;MAEZ,oB;MAAtB,mBAAc,CAAd,8B;QACI,IA  
AI,CAAC,QAAS,UAAAd,C;UAAyB,K;QACzB,mBAAY,OAAZ,IAAqB,QAAS,O;;MAGIC,wBAAQ,QAAS,KAAj  
B,I;K;0CAGJ,oB;MACI,IAAI,QAAS,UAAb,C;QAAwB,OAAO,K;MAC/B,sBAAE,IAAK,KAAL,GAAy,QAAS,K

AArB,IAAf,C;MACA,8BAAtLgD,mBAAY,cAsLvB,SAAtLuB,IAAZ,CAsLhD,EAA4C,QAA5C,C;MACA,OAAO,I;K  
;0CAGX,2B;MACI,oCAAA,4BAAmB,KAAAnB,EAA0B,SAA1B,C;MAEb,IAAI,QAAS,UAAb,C;QACI,OAAO,K;a  
ACJ,IAAI,UAAAS,SAAb,C;QACH,OAAO,oBAAO,QAAP,C;OAGX,sBAAe,IAAK,KAAL,GAAY,QAAS,KAArB,I  
AAf,C;MAEA,WArMgD,mBAAY,cAqMnC,SArMmC,IAAZ,C;MASMhD,oBAAtMgD,mBAAY,cAsM1B,KAtM0B,  
IAAZ,C;MAuMhD,mBAAmB,QAAS,K;MAE5B,IAAI,QAAS,SAAD,GAAQ,CAAR,IAAe,CAA3B,C;QAGI,kBA  
AkB,cAAO,YAAP,I;QAEIB,IAAI,iBAAiB,WAArB,C;UACI,IAAI,eAAe,CAAnB,C;Y9J0mBZ,U8JzmBgB,kB9Jy  
mBhB,E8JzmBqC,kB9JymBrC,E8JzmBkD,W9JymBID,E8JzmB+D,W9JymB/D,E8JzmBqE,a9JymBrE,C;;Y8Jvmb  
gB,4BAAe,kBAAY,OAA3B,I;YACA,sBAAsB,gBAAGB,WAAhB,I;YACtB,kBAAkB,kBAAY,OAAZ,GAAMB,W  
AAnB,I;YAEIB,IAAI,eAAe,eAAnB,C;c9JmmBhB,U8JlmBoB,kB9JkmBpB,E8JlmByC,kB9JkmBzC,E8JlmBsD,W  
9JkmBtD,E8JlmBmE,W9JkmBnE,E8JlmByE,a9JkmBzE,C;;cAAA,U8JhmBoB,kB9JgmBpB,E8JhmByC,kB9JgmB  
zC,E8JhmBsD,W9JgmBtD,E8JhmBmE,W9JgmBnE,E8JhmByE,cAAO,WAAP,I9JgmBzE,C;cAAA,U8JlBoB,kB9  
J+lBpB,E8JlByC,kB9J+lBzC,E8JlBsD,C9J+lBtD,E8JlByD,cAAO,WAAP,I9J+lBzD,E8JlB6E,a9J+lB7E,C;;;UA  
AA,U8J3lBY,kB9J2lBZ,E8J3lBiC,kB9J2lBjC,E8J3lB8C,W9J2lB9C,E8J3lB2D,W9J2lB3D,E8J3lBiE,kBAAY,O9J  
2lB7E,C;U8J1lBY,IAAI,gBAAGB,aAApB,C;Y9J0lBZ,U8JzlBgB,kB9JylBhB,E8JzlBqC,kB9JylBrC,E8JzlBkD,kB  
AAY,OAAZ,GAAMB,YAAnB,I9JylBID,E8JzlBmF,C9JylBnF,E8JzlBsF,a9JylBtF,C;;YAAA,U8JvlBgB,kB9JulBh  
B,E8JvlBqC,kB9JulBrC,E8JvlBkD,kBAAY,OAAZ,GAAMB,YAAnB,I9JulBID,E8JvlBmF,C9JulBnF,E8JvlBsF,Y9  
JulBtF,C;YAAA,U8JtlBgB,kB9JslBhB,E8JtlBqC,kB9JslBrC,E8JtlBkD,C9JslBID,E8JtlBqD,Y9JslBrD,E8JtlBmE,a  
9JslBnE,C;;;Q8JnlBQ,cAAO,W;QACP,8BAAuB,mBAAY,gBAAGB,YAAhB,IAAZ,CAAvB,EAAkE,QAAIE,C;;Q  
AIA,2BAA2B,gBAAGB,YAAhB,I;QAE3B,IAAI,gBAAGB,IAApB,C;UACI,IAAI,QAAO,YAAP,SAAuB,kBAAY,  
OAAvC,C;Y9J2kBZ,U8J1kBgB,kB9J0kBhB,E8J1kBqC,kB9J0kBrC,E8J1kBkD,oB9J0kBID,E8J1kBwE,a9J0kBxE,  
E8J1kBuF,I9J0kBvF,C;;Y8JxkBgB,IAAI,wBAAwB,kBAAY,OAAxC,C;c9JwkBhB,U8JvkBoB,kB9JukBpB,E8Jvk  
ByC,kB9JukBzC,E8JvkBsD,uBAAuB,kBAAY,OAAAnC,I9JukBtD,E8JvkB+F,a9JukB/F,E8JvkB8G,I9JukB9G,C;;c  
8JrkBoB,mBAAmB,OAAO,YAAP,GAAsB,kBAAY,OAAIC,I;c9JqkBvC,U8JpkBoB,kB9JokBpB,E8JpkByC,kB9J  
okBzC,E8JpkBsD,C9JokBtD,E8JpkByD,OAAO,YAAP,I9JokBzD,E8JpkB8E,I9JokB9E,C;cAAA,U8JnkBoB,kB9J  
mkBpB,E8JnkByC,kB9JmkBzC,E8JnkBsD,oB9JmkBtD,E8JnkB4E,a9JmkB5E,E8JnkB2F,OAAO,YAAP,I9JmkB3  
F,C;;;UAAA,U8JlJBY,kB9J+lBZ,E8JlJBiC,kB9J+lBjC,E8JlJb8C,Y9J+lB9C,E8JlJb4D,C9J+lB5D,E8JlJb+D,I9J+l  
B/D,C;U8J9jBY,IAAI,wBAAwB,kBAAY,OAAxC,C;Y9J8jBZ,U8J7jBgB,kB9J6jBhB,E8J7jBqC,kB9J6jBrC,E8J7j  
BkD,uBAAuB,kBAAY,OAAAnC,I9J6jBID,E8J7jB2F,a9J6jB3F,E8J7jB0G,kBAAY,O9J6jBtH,C;;YAAA,U8J3jBgB,  
kB9J2jBhB,E8J3jBqC,kB9J2jBrC,E8J3jBkD,C9J2jBID,E8J3jBqD,kBAAY,OAAZ,GAAMB,YAAnB,I9J2jBrD,E8J  
3jBsF,kBAAY,O9J2jBlG,C;YAAA,U8J1jBgB,kB9J0jBhB,E8J1jBqC,kB9J0jBrC,E8J1jBkD,oB9J0jBID,E8J1jBwE  
,a9J0jBxE,E8J1jBuF,kBAAY,OAAZ,GAAMB,YAAnB,I9J0jBvF,C;;;Q8JvjBQ,8BAAuB,aAAvB,EAAcS,QAAtC,  
C;;MAGJ,OAAO,I;K;uCAGX,iB;MACI,oCAAA,2BAAkB,KAAIB,EAAyB,SAAzB,C;MAjRN,Q;MAmRP,OAnR  
O,2BAQyC,mBAAY,cA2Q3B,KA3Q2B,IAAZ,CARzC,4D;K;uCAAsRX,0B;MACI,oCAAA,2BAAkB,KAAIB,EAAy  
B,SAAzB,C;MAEb,oBAjRgD,mBAAY,cAiR1B,KAjR0B,IAAZ,C;MARzC,Q;MA0RP,iBA1RO,2BA0RsB,aA1Rt  
B,4D;MA2RP,mBAAY,aAAZ,IAA6B,O;MAE7B,OAAO,U;K;0CAGX,mB;MAAoD,0BAAQ,OAAR,MAAoB,E;K  
;yCAExE,mB;MAIsB,IAIA,IAJA,EAIuB,M;MAPzC,WA3RgD,mBAAY,cA2RnC,SA3RmC,IAAZ,C;MA6RhD,IA  
AI,cAAO,IAAX,C;QACI,iBAAc,WAAAd,UAAyB,IAAzB,U;UACI,IAAI,gBAAW,mBAAY,KAAZ,CAAX,CAAJ,C  
;YAAmC,OAAO,QAAQ,WAAR,I;;aAE3C,IAAI,eAAQ,IAAZ,C;QACW,kB;QAAuB,SAAZ,kBAAY,O;QAAR,c,q  
D;UACI,IAAI,gBAAW,mBAAY,OAAZ,CAAX,CAAJ,C;YAAmC,OAAO,UAAQ,WAAR,I;;QAE9C,mBAAc,CA  
Ad,YAAsB,IAAtB,Y;UACI,IAAI,gBAAW,mBAAY,OAAZ,CAAX,CAAJ,C;YAAmC,OAAO,UAAQ,kBAAY,OA  
ApB,GAA2B,WAA3B,I;;OAIID,OAAO,E;K;6CAGX,mB;MAIsC,UAOJ,MAPI,EAOa,M;MAV/C,WA9SgD,mBA  
AY,cA8SnC,SA9SmC,IAAZ,C;MAGThD,IAAI,cAAO,IAAX,C;QACKc,kB;QAA9B,iBAAc,OAAO,CAAP,IAAd,y  
B;UACI,IAAI,gBAAW,mBAAY,KAAZ,CAAX,CAAJ,C;YAAmC,OAAO,QAAQ,WAAR,I;;aAE3C,IAAI,cAAO,I  
AAX,C;QACH,mBAAc,OAAO,CAAP,IAAd,aAA8B,CAA9B,Y;UACI,IAAI,gBAAW,mBAAY,OAAZ,CAAX,CA  
AJ,C;YAAmC,OAAO,UAAQ,kBAAY,OAAP,GAA2B,WAA3B,I;;QAEpB,uBAAZ,kBAAY,C;QAAiB,oB;QAA  
3C,wD;UACI,IAAI,gBAAW,mBAAY,OAAZ,CAAX,CAAJ,C;YAAmC,OAAO,UAAQ,WAAR,I;;OAIID,OAAO,E  
;K;wCAGX,mB;MACI,YAAY,mBAAQ,OAAR,C;MACZ,IAAI,UAAAS,EAAb,C;QAAiB,OAAO,K;MACxB,sBAA  
S,KAAT,C;MACA,OAAO,I;K;4CAGX,iB;MACI,oCAAA,2BAAkB,KAAIB,EAAyB,SAAzB,C;MAEb,IAAI,UAA

S,sBAAb,C;QACI,OAAO,iB;aACJ,IAAI,UAAS,CAAb,C;QACH,OAAO,kB;OAGX,oBAhVgD,mBAAY,cAgV1B,KAhV0B,IAAZ,C;MARzC,Q;MAyVP,cAzVO,2BAyVmB,aAzVnB,4D;MA2VP,IAAI,QAAQ,aAAS,CAArB,C;QAEI,IAAI,iBAAiB,WAArB,C;U9JoeR,U8JneY,kB9JmeZ,E8JneiC,kB9JmeJ,C,E8Jne8C,cAAO,CAAP,I9Jme9C,E8JnewD,W9JmexD,E8Jne8D,a9Jme9D,C;;UAAA,U8JjeY,kB9JieZ,E8JjeiC,kB9JiejC,E8Jje8C,C9Jie9C,E8JjeiD,C9JiejD,E8JjeoD,a9JjepD,C;U8JheY,mBAAY,CAAZ,IAAiB,mBAAY,kBAAY,OAAZ,GAAmB,CAAnB,IAAZ,C;U9Jge7B,U8J/dY,kB9J+dZ,E8J/diC,kB9J+djC,E8J/d8C,cAAO,CAAP,I9J+d9C,E8J/dwD,W9J+dxD,E8J/d8D,kBAAY,OAAZ,GAAmB,CAAnB,I9J+d9D,C;;Q8J5dQ,mBAAY,WAAZ,IAAoB,I;QACpB,cAAO,mBAAY,WAAZ,C;;QAGP,wBAjW4C,mBAAY,cAiWlB,sBAjWkB,IAAZ,C;QAmW5C,IAAI,iBAAiB,iBAArB,C;U9JsdR,U8JrdY,kB9JqdZ,E8JrdiC,kB9JqjC,E8Jrd8C,a9Jqd9C,E8Jrd6D,gBAAGB,CAAhB,I9Jqd7D,E8JrdgF,oBAAoB,CAApB,I9JqdhF,C;;UAAA,U8JndY,kB9JmdZ,E8JndiC,kB9JmdjC,E8Jnd8C,a9Jmd9C,E8Jnd6D,gBAAGB,CAAhB,I9Jmd7D,E8JndgF,kBAAY,O9Jmd5F,C;U8JldY,mBAAY,kBAAY,OAAZ,GAAmB,CAAnB,IAAZ,IAAoC,mBAAY,CAAZ,C;U9JkdhD,U8JjdY,kB9JidZ,E8JjdiC,kB9JidjC,E8Jjd8C,C9Jid9C,E8JjdiD,C9JidjD,E8JjdoD,oBAAoB,CAApB,I9JidpD,C;;Q8J9cQ,mBAAY,iBAAZ,IAAiC,I;;MAErC,wBAAQ,CAAR,I;MAEA,OAAO,O;K;6CAGX,oB;MAAkE,OB;;QAa5C,wD;QART,aAAL,IAAK,U;QAAL,Y;UAA8B,SAAZ,kB7K6wOnB,YAAQ,C;S6K7wOX,W;UACI,yBAAO,K;UAAp,2B;SAEJ,WA1XgD,mBAAY,cA0XnC,SA1XmC,IAAZ,C;QA2XhD,cAAc,W;QACd,eAAe,K;QAEf,IAAI,cAAO,IAAX,C;UACI,iBAAc,WAAd,UAAyB,IAAzB,U;YACI,cAAc,mBAAY,KAAZ,C;YAGd,IAjBsE,CAAU,wBAiBIE,0EAjBkE,CAiBhF,C;cACI,mBAAY,gBAAZ,EAAY,wBAAZ,YAAyB,O;;cAEzB,WAAW,I;;UAGP,OAAZ,kBAAY,EAkK,IAAL,EAAW,OAAX,EAAoB,IAApB,C;;UAGE,oB;UAAuB,SAAZ,kBAAY,O;UAArC,uD;YACI,gBAAc,mBAAY,OAAZ,C;YACd,mBAAY,OAAZ,IAAqB,I;YAGrB,IA/BsE,CAAU,wBA+BIE,kFA/BkE,CA+BhF,C;cACI,mBAAY,gBAAZ,EAAY,wBAAZ,YAAyB,S;;cAEzB,WAAW,I;;UAGnB,UAAU,mBAAY,OAAZ,C;UAEV,mBAAc,CAAd,YAAsB,IAAtB,Y;YACI,gBAAc,mBAAY,OAAZ,C;YACd,mBAAY,OAAZ,IAAqB,I;YAGrB,IA5CsE,CAAU,wBA4CIE,kFA5CKE,CA4ChF,C;cACI,mBAAY,OAAZ,IAAuB,S;cACvB,UAAU,mBAAY,OAAZ,C;;cAEV,WAAW,I;;;QAIvB,IAAI,QAAJ,C;UACI,YAAO,mBAAY,UAAU,WAAV,IAAZ,C;QAEX,yBAAO,Q;;MAvDuD,6B;K;6CAEIE,oB;MAAkE,OB;;QAW5C,wD;QART,aAAL,IAAK,U;QAAL,Y;UAA8B,SAAZ,kB7K6wOnB,YAAQ,C;S6K7wOX,W;UACI,yBAAO,K;UAAp,2B;SAEJ,WA1XgD,mBAAY,cA0XnC,SA1XmC,IAAZ,C;QA2XhD,cAAc,W;QACd,eAAe,K;QAEf,IAAI,cAAO,IAAX,C;UACI,iBAAc,WAAd,UAAyB,IAAzB,U;YACI,cAAc,mBAAY,KAAZ,C;YAGd,IAf+E,wBAejE,0EAfiE,CAe/E,C;cACI,mBAAY,gBAAZ,EAAY,wBAAZ,YAAyB,O;;cAEzB,WAAW,I;;UAGP,OAAZ,kBAAY,EAkK,IAAL,EAAW,OAAX,EAAoB,IAApB,C;;UAGE,oB;UAAuB,SAAZ,kBAAY,O;UAArC,uD;YACI,gBAAc,mBAAY,OAAZ,C;YACd,mBAAY,OAAZ,IAAqB,I;YAGrB,IA7B+E,wBA6BjE,kFA7BiE,CA6B/E,C;cACI,mBAAY,gBAAZ,EAAY,wBAAZ,YAAyB,S;;cAEzB,WAAW,I;;UAGnB,UAAU,mBAAY,OAAZ,C;UAEV,mBAAc,CAAd,YAAsB,IAAtB,Y;YACI,gBAAc,mBAAY,OAAZ,C;YACd,mBAAY,OAAZ,IAAqB,I;YAGrB,IA1C+E,wBA0CjE,kFA1CiE,CA0C/E,C;cACI,mBAAY,OAAZ,IAAuB,S;cACvB,UAAU,mBAAY,OAAZ,C;;cAEV,WAAW,I;;;QAIvB,IAAI,QAAJ,C;UACI,YAAO,mBAAY,UAAU,WAAV,IAAZ,C;QAEX,yBAAO,Q;;MArDuD,6B;K;2CAEIE,qB;MASSb,IAII,IAJJ,EAKM,MALN,EAaA,MABa,EAuB,MABvB,EAKBI,MAIBJ,EAmBM,MANBN,EA+BI,M;MAvCb,aAAL,IAAK,U;MAAL,Y;QAA8B,SAAZ,kB7K6wOnB,YAAQ,C;O6K7wOX,W;QACI,OAAO,K;MAEX,WA1XgD,mBAAY,cA0XnC,SA1XmC,IAAZ,C;MA2XhD,cAAc,W;MACd,eAAe,K;MAEf,IAAI,cAAO,IAAX,C;QACI,iBAAc,WAAd,UAAyB,IAAzB,U;UACI,cAAc,mBAAY,KAAZ,C;UAGd,IAAI,UAAU,0EAAV,CAAJ,C;YACI,mBAAY,gBAAZ,EAAY,wBAAZ,YAAyB,O;;YAEzB,WAAW,I;;QAGP,OAAZ,kBAAY,EAkK,IAAL,EAAW,OAAX,EAAoB,IAApB,C;;QAGE,oB;QAAuB,SAAZ,kBAAY,O;QAARc,uD;UACI,gBAAc,mBAAY,OAAZ,C;UACd,mBAAY,OAAZ,IAAqB,I;UAGrB,IAAI,UAAU,kFAAV,CAAJ,C;YACI,mBAAY,gBAAZ,EAAY,wBAAZ,YAAyB,S;;YAEzB,WAAW,I;;QAGnB,UAAU,mBAAY,OAAZ,C;QAEV,mBAAc,CAAd,YAAsB,IAAtB,Y;UACI,gBAAc,mBAAY,OAAZ,C;UACd,mBAAY,OAAZ,IAAqB,I;UAGrB,IAAI,UAAU,kFAAV,CAAJ,C;YACI,mBAAY,OAAZ,IAAuB,S;YACvB,UAAU,mBAAY,OAAZ,C;;YAEV,WAAW,I;;;MAIvB,IAAI,QAAJ,C;QACI,YAAO,mBAAY,UAAU,WAAV,IAAZ,C;MAEX,OAAO,Q;K;iCAGX,Y;MACI,WA7agD,mBAAY,cA6anC,SA7amC,IAAZ,C;MA8ahD,IAAI,cAAO,IAAX,C;QACgB,OAAZ,kBAAY,EAkK,IAAL,EA AW,WAAx,EAaiB,IAAjB,C;;QACT,IvKtS6C,CAAC,cuKsS9C,C;UACS,OAAZ,kBAAY,EAkK,IAAL,EA AW,WAAx,EAaiB,kBAAY,OAA7B,C;UACA,OAAZ,kBAAY,EAkK,IAAL,EA AW,CAAX,EAAC,IAAd,C;;MAEhB,cAAO,C;MACP,YAAO,C;K;2CAGX,iB;MAGe,IAAC,IAAD,EAcJ,M;MAfP,WACW,eAAC,OAAI,KAAM,OAA

N,IAAc,SAAlB,GAAwB,KAAxB,GAAmC,aAAa,KAAb,EAAoB,SAApB,CAApC,uB;MAEX,WA7bgD,mBAAY,cA6bnC,SA7bmC,IAAZ,C;MA8bhD,IAAI,cAAO,IAAX,C;Q9J2XJ,U8J1XQ,kB9J0XR,E8J1X6B,I9J0X7B,EAD+F,CAC/F,E8J1XgD,W9J0XhD,E8J1XiE,I9J0XjE,C;;Q8JzXW,IvKtT6C,CAAC,cuKsT9C,C;U9JyXX,U8JxXQ,kB9JwXR,E8JxX6B,I9JwX7B,E8JxXuD,C9JwXvD,E8JxXuE,W9JwXvE,E8JxXwF,kBAAY,O9JwXpG,C;UAAA,U8JvXQ,kB9JuXR,E8JvX6B,I9JuX7B,E8JvXuD,kBAAY,OAAZ,GAAmB,WAAAnB,I9JuXvD,E8JvX6F,C9JuX7F,E8JvX2G,I9JuX3G,C;;M8JrXI,IAAI,IAAK,OAAL,GAAY,SAAhB,C;QACI,KAAK,SAAL,IAAa,I;OAIjB,OAAO,qD;K;mCAGX,Y;MAEI,OAAO,qBAAQ,gBAAmB,SAAnB,OAAR,C;K;+CAGX,iB;MAC0D,4BAAQ,KAAR,C;K;+CAC1D,Y;MAA0C,qB;K;IAE1C,gC;MAAA,oC;MACI,0BpHriBuC,E;MoHsiBvC,sBAAiC,U;MACjC,4BAAuC,E;K;yDAEvC,oC;MAEI,kBAAkB,eAAe,eAAGB,CAA/B,K;MACIB,IAAI,eAAc,WAAAd,QAA4B,CAAhC,C;QACI,cAAc,W;MACIB,IAAI,eAAc,UAAAd,QAA6B,CAAjC,C;QACI,cAAkB,cAAc,UAAIB,GAAgC,UAAhC,GAAmD,U;MACrE,OAAO,W;K;;IAZf,4C;MAAA,2C;QAAA,0B;OAAA,oC;K;qDAgBA,qB;MAEI,WAVEGd,mBAAY,cAuenC,SAvemC,IAAZ,C;MAwehD,WAAe,kBAaA,cAAO,IAAxB,GAA8B,WAA9B,GAAwC,cAAO,kBAAY,OAAAnB,I;MACnD,UAAU,IAAV,EAAgB,cAAhB,C;K;;IA5iBJ,iD;MAAA,oD;MAGwC,+B;MApB5C,sB;MAqBsB,Q;MACV,wBAAmB,CAAnB,C;QAAwB,4D;WACxB,sBAAkB,CAAlB,C;QAAuB,uBAaA,eAAb,O;;QACf,MAAM,gCAAYB,uBAAoB,eAA7C,C;MAHIB,0B;MAJJ,Y;K;IAWA,kC;MAAA,oD;MAGoB,+B;MA/BxB,sB;MAGCQ,sBAAc,qD;MAJIB,Y;K;IAOA,4C;MAAA,oD;MAG2C,+B;MATc/C,sB;MAuQC,sBxJrB8D,YwJqBhD,QxJrBgD,C;MwJsB9D,aAAO,mBAAY,O;MACnB,IAAI,mB7K+qPD,YAAQ,C6K/qPX,C;QAA2B,sBAAc,qD;MAN7C,Y;K;IC5BJ,4B;MAMoB,Q;M9KghqBA,U;MADhB,UAAe,C;MACf,uD;QAAgB,cAAhB,iB;QACI,YAAgB,O8KlhqBiB,O9KkhqBjC,I;;M8KlhqBJ,aAAa,iB9KohqBN,G8KphqBM,C;MACb,wBAAGB,SAAhB,gB;QAAgB,gBAAA,SAAhB,M;QACW,SAAP,MAAO,EAAO,SAAP,C;;MAEX,OAAO,M;K;IAGX,0B;MASiB,Q;MAFb,YAAy,iBAaA,gBAAb,C;MACZ,YAAy,iBAaA,gBAAb,C;MACZ,wBAaA,SAAb,gB;QAAa,WAAA,SAAb,M;QACI,KAAM,WAAI,IAAK,MAAT,C;QACN,KAAM,WAAI,IAAK,OAAT,C;;MAEV,OAAO,UAAAS,KAAT,C;K;gGAGX,qB;MAWW,4B;MAAA,U;QAAqB,OAAL,S9K0qPhB,YAAQ,C;O8K1qPf,W;K;oFAGJ,mC;MAUI,O9K6pPO,qBAAQ,C8K7pPf,GAAe,cAAf,GAAmC,S;K;IAGvC,iD;MAMI,IAAI,cAAS,KAAb,C;QAAoB,OAAO,I;MAC3B,IAAI,qBAAgB,aAAhB,IAAiC,SAAK,OAAL,KAAa,KAAM,OAAXD,C;QAA8D,OAAO,K;MAErE,4C;QACI,SAAS,UAAK,CAAL,C;QACT,SAAAS,MAAM,CAAN,C;QAET,IAAI,OAAO,EAAX,C;UACI,Q;eACG,IAAI,cAAc,UAAIB,C;UACH,OAAO,K;SAIP,0BAAsB,kBAAtB,C;UAA4C,IAAI,CAAI,kBAAH,EAAG,EAakB,EAAlB,CAAR,C;YAA+B,OAAO,K;eACIF,8BAAsB,sBAAtB,C;UAA4C,IAAI,CAAI,cAAH,EAAG,EAAC,EAAd,CAAR,C;YAA2B,OAAO,K;eAC9E,+BAAsB,uBAAtB,C;UAA4C,IAAI,CAAI,cAAH,EAAG,EAAC,EAAd,CAAR,C;YAA2B,OAAO,K;eAC9E,6BAAsB,qBAAtB,C;UAA4C,IAAI,CAAI,cAAH,EAAG,EAAC,EAAd,CAAR,C;YAA2B,OAAO,K;eAC9E,8BAAsB,sBAAtB,C;UAA4C,IAAI,CAAI,cAAH,EAAG,EAAC,EAAd,CAAR,C;YAA2B,OAAO,K;eAC9E,+BAAsB,uBAAtB,C;UAA4C,IAAI,CAAI,cAAH,EAAG,EAAC,EAAd,CAAR,C;YAA2B,OAAO,K;eAC9E,8BAAsB,sBAAtB,C;UAA4C,IAAI,CAAI,cAAH,EAAG,EAAC,EAAd,CAAR,C;YAA2B,OAAO,K;eAC9E,iCAAsB,yBAAtB,C;UAA4C,IAAI,CAAI,cAAH,EAAG,EAAC,EAAd,CAAR,C;YAA2B,OAAO,K;eAE9E,qCAAsB,6BAAtB,C;UAA4C,IAAI,CAAI,gBAAH,EAAG,EAAC,EAAd,CAAR,C;YAA2B,OAAO,K;eAC9E,sCAAsB,8BAAtB,C;UAA4C,IAAI,CAAI,gBAAH,EAAG,EAAC,EAAd,CAAR,C;YAA2B,OAAO,K;eAC9E,oCAAsB,4BAAtB,C;UAA4C,IAAI,CAAI,gBAAH,EAAG,EAAC,EAAd,CAAR,C;YAA2B,OAAO,K;eAEtE,IAAI,YAAM,EAAN,CAAJ,C;UAAc,OAAO,K;;MAIrC,OAAO,I;K;IAGX,4C;MAKI,IAAI,iBAAJ,C;QAAkB,OAAO,M;MACzB,aAAa,CAAK,eAAL,gBAAK,EAAa,SAAb,CAAL,GAA6C,CAA7C,QAAiD,CAAjD,I;MvC6SkB,kBAAxB,mBuC5SY,MvC4SZ,C;MuC3SH,oDxK5BgD,gBwK4BhD,C;MADJ,O7JnC O,WsH+U6C,W;K;IuCvSxD,mE;MAEI,IAAY,SAAR,0BAAJ,C;QACI,MAAO,gBAAO,OAAP,C;QACP,M;OAEJ,SAAU,WAAI,SAAJ,C;MACV,MAAO,gBAAO,EAAP,C;MAEP,4C;QACI,IAAI,MAAK,CAAT,C;UACI,MAAO,gBAAO,IAAP,C;SAEX,cAAc,UAAK,CAAL,C;QAEV,IADE,OACF,S;UAAmB,MAAO,gBAAO,MAAP,C;aAC1B,mBAFE,OAEF,E;UAA2B,4BAAR,OAAQ,EAA4B,MAA5B,EAAoC,SAAP,C,C;aAC3B,uBAHE,OAGF,E;UAAmB,MAAO,gBAAE,gBAAR,OAAQ,CAAF,C;aAC1B,wBAJE,OAIIF,E;UAAmB,MAAO,gBAAE,gBAAR,OAAQ,CAAF,C;aAC1B,sBALE,OAKF,E;UAAmB,MAAO,gBAAE,gBAAR,OAAQ,CAAF,C;aAC1B,uBANE,OAMF,E;UAAmB,MAAO,gBAAE,gBAAR,OAAQ,CAAF,C;aAC1B,wBAPE,OAOFF,E;UAAmB,MAAO,gBAAE,gBAAR,OAAQ,

CAAf,C;aAC1B,yBARE,OAQF,E;UAAmB,MAAO,gBA Ae,gBAAR,OAAQ,CAAf,C;aAC1B,uBATE,OASF,E;UA  
AmB,MAAO,gBA Ae,gBAAR,OAAQ,CAAf,C;aAC1B,0BAVE,OAUF,E;UAAmB,MAAO,gBA Ae,gBAAR,OAA  
Q,CAAf,C;aAE1B,kBAZE,OAYF,c;UAAmB,MAAO,gBA Ae,kBAAR,OAAQ,CAAf,C;aAC1B,kBA bE,OAaF,e;U  
AAmB,MAAO,gBA Ae,kBAAR,OAAQ,CAAf,C;aAC1B,kBA dE,OAcf,a;UAAmB,MAAO,gBA Ae,kBAAR,OAA  
Q,CAAf,C;aAC1B,kBA fE,OAeF,c;UAAmB,MAAO,gBA Ae,kBAAR,OAAQ,CAAf,C;;UAEP,MAAO,gBAAO,OA  
AQ,WAAf,C;;MAIIC,MAAO,gBAAO,EAAP,C;MACP,SAAU,kBAAmB,iBAAV,SAAU,CAAnB,C;K;ICpJd,uC;  
MAIqD,+CAAwC,iBAAO,CAA/C,IAAoD,mC;K;IAEzG,4D;MAWQ,kBADE,SACF,O;QADJ,OACc,S;WACV,kB  
AFE,SAEF,c;QAEQ,yCAAwB,MAAO,KAAP,GAAC,CAAtC,C;UAJZ,OAIuD,S;;UAJvD,OAK6B,mBAAL,SAAK  
,CAAT,GAA+B,sBAA/B,GAAgD,S;;QALpE,OAoGB,oCAAJ,GAA0C,sBAA1C,GAA2D,mB;K;IAG3E,gD;MAW  
Q,kBADE,SACF,O;QADJ,OACc,S;WACV,kBAFE,SAEF,c;QAFJ,OAE8B,mBAAL,SAAK,CAAT,GAA+B,sBAA  
/B,GAAgD,S;;QAFrE,OAGgB,oCAAJ,GAA0C,sBAA1C,GAA2D,mB;K;IAG3E,kD;MAKI,OAAI,oCAAJ,GAA0C  
,sBAA1C,GAA2D,oB;K;IAE/D,kD;MAKI,OAAI,oCAAJ,GAA0C,oBAA1C,GAA2D,iB;K;IzKnD/D,yB;MAAA,6  
B;K;sCACI,Y;MAAkC,Y;K;0CACIC,Y;MAAsC,Y;K;wCACtC,Y;MAAgC,Q;K;4CACChC,Y;MAAoC,S;K;mCACp  
C,Y;MAA+B,MAAM,6B;K;uCACrC,Y;MAAmC,MAAM,6B;K;;IAN7C,qC;MAAA,oC;QAAA,mB;OAAA,6B;K;  
IASA,qB;MAAA,yB;MACI,+C;K;iCAEA,iB;MAA4C,qCAAoB,KAAM,U;K;mCACtE,Y;MAA+B,Q;K;mCAC/B,  
Y;MAAkC,W;K;iFAEX,Y;MAAQ,Q;K;kCAC/B,Y;MAAkC,W;K;yCACIC,mB;MAAmD,Y;K;8CACnD,oB;MAA  
mE,OAAA,QAAS,U;K;sCAE5E,iB;MAAwC,MAAM,8BAA0B,iDAA8C,KAA9C,MAA1B,C;K;wCAC9C,mB;M  
AA8C,S;K;4CAC9C,mB;MAAkD,S;K;mCAEID,Y;MAA6C,kC;K;uCAC7C,Y;MAAqD,kC;K;+CACrD,iB;MACI,  
IAAI,UAAS,CAAb,C;QAAgB,MAAM,8BAA0B,YAAS,KAAnC,C;MACtB,OAAO,2B;K;0CAGX,8B;MACI,IAAI  
,cAAa,CAAb,IAAkB,YAAW,CAAjC,C;QAAoC,OAAO,I;MAC3C,MAAM,8BAA0B,gBAAa,SAAb,mBAAkC,O  
AA5D,C;K;wCAGV,Y;MAAiC,8B;K;;IA5BrC,iC;MAAA,gC;QAAA,e;OAAA,yB;K;IA+BA,iC;MAA8D,6BAAk  
B,SAAlB,EAAoC,KAApC,C;K;IAE5B,8C;MAAC,oB;MAA0B,0B;K;yFACIC,Y;MAAQ,OAAA,WAAO,O;K;0C  
ACtC,Y;MAAkC,OAAA,WNqqP3B,YAAQ,C;K;iDMpqPf,mB;MAA6C,OAAO,SAAP,WAAO,EAAS,OAAT,C;K  
;SDACpD,oB;MAAsE,c;;Qc4nDtD,Q;QADhB,IAAI,cd3nDyD,Qc2nDzD,iBd3nDyD,Qc2nDnC,UAA1B,C;UAAqC  
,aAAO,I;UAAP,e;SACrB,Od5nD6C,Qc4nD7C,W;QAAhB,OAAgB,cAAhB,C;UAAgB,yB;UAAM,IAAI,Cd5nDkD  
,oBc4nDvC,Od5nDuC,Cc4nDtD,C;YAAyB,aAAO,K;YAAP,e;;QAC/C,aAAO,I;;Md7nDsD,iB;K;2CAC7D,Y;MA  
AuC,OAAO,qBAAP,WAAO,C;K;0CAC9C,Y;MAC+C,gBAAP,W;MAAA,OAAwB,cAAxB,GegKpC,SfhKoC,Gek  
KpC,SN83BoB,Q;K;;IT7hC5B,qB;MAIsC,8B;K;IAEtC,4B;MAIqD,OAAI,QAAS,OAAT,GAAgB,CAAPB,GAAg  
C,OAAT,QAAS,CAAhC,GAA8C,W;K;mFAEnG,yB;MAAA,qD;MAAA,mB;QAK0C,kB;O;KAL1C,C;+FAOA,y  
B;MAAA,+D;MAAA,mB;QAMwD,uB;O;KANxD,C;2FAQA,yB;MAAA,+D;MAAA,mB;QAMoD,uB;O;KANpD,  
C;IAQA,mC;MAKI,OAAI,QAAS,OAAT,KAAiB,CAArB,GAAwB,gBAAxB,GAAyC,iBAAU,sBAAkB,QAAIB,E  
AAwC,IAAxC,CAAV,C;K;IAE7C,iC;MAKI,OAAI,QAAS,OAAT,KAAiB,CAArB,GAAwB,gBAAxB,GAAyC,iB  
AAU,sBAAkB,QAAIB,EAAwC,IAAxC,CAAV,C;K;IAE7C,gC;MAI2D,OAAI,eAAJ,GAAqB,OAAO,OAAP,CAA  
rB,GAA0C,W;K;IAErG,mC;MAImE,OAAS,cAAT,QAAS,C;K;gFAE5E,yB;MAAa,gE;MAbA,6B;QAYBI,WAAW  
,eAduE,IAcvE,C;QWCX,iBAAc,CAAd,UXfkF,IWefU;UXA6B,eAf2D,IAevD,CWctB,KXDsb,CAAJ,C;;QAFyC  
,OAGB/D,I;O;KA3BX,C;8FAaA,yB;MAAA,gE;MAAA,6B;QAYI,WAAW,eAAa,IAAb,C;QWCX,iBAAc,CAAd,U  
XAO,IWAP,U;UXA6B,eAAI,KWctB,KXDsb,CAAJ,C;;QAC7B,OAAO,I;O;KAdX,C;wFAiBA,yB;Me1FA,+D;M  
f0FA,gC;QetF0B,gBAAf,gB;QfsGkB,aW3FzB,W;QX2FA,OW1FO,SIZoC,Q;O;KfsF/C,C;yFAwBA,yB;Me3GA,4  
E;MAAA,gE;Mf2GA,0C;QevGI,qBf2HyB,Qe3HzB,C;QAC8B,gBAAvB,ef0HkB,Qe1HIB,C;Qf0H4B,aWvHnC,W;  
QXuHA,OWtHO,SIJ4C,Q;O;KfsGvD,C;IAiCI,mC;MAAQ,uBAAG,iBAAO,CAAP,IAAH,C;K;IAQR,qC;MAAQ,  
OAAA,SAAK,KAAL,GAAY,CAAZ,I;K;4FAEZ,qB;MAK4D,QAAC,mB;K;kGAE7D,qB;MAWI,OAAO,qBAAgB  
,SAAK,U;K;sFAGhC,yB;MAAA,qD;MAAA,4B;QAKgE,uCAAQ,W;O;KALxE,C;sFAOA,yB;MAAA,qD;MAAA,  
4B;QAKoD,uCAAQ,W;O;KAL5D,C;sFAOA,mC;MASI,OAAI,mBAAJ,GAAe,cAAf,GAAMC,S;K;4FAGvC,+B;  
MAQoH,OAAA,SAAK,qBAAY,QAAZ,C;K;IAGzH,uC;MAK+E,kBAAhB,0B;MAAwB,+B;MAAxB,OW5MpD,  
W;K;IX+MX,yC;MAAkD,QAAM,cAAN,C;aAC9C,C;UAD8C,OACzC,W;aACL,C;UAF8C,OAeZC,OAAO,sBAA  
K,CAAL,CAAP,C;gBAFyC,OAGtC,S;;K;IAGZ,8D;MAGkE,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,c;M  
ACjG,WAAW,cAAX,EAAiB,SAAjB,EAA4B,OAA5B,C;MAEA,UAAU,S;MACV,WAAW,UAAU,CAAV,I;MAE  
X,OAAO,OAAO,IAAd,C;QACI,UAAW,GAAY,GAAN,IAAM,KAAK,C;QAC5B,aAAa,sBAAI,GAAJ,C;QACb,U

AAU,cAAc,MAAd,EAA5B,OAAtB,C;QAEV,IAAI,MAAM,CAAV,C;UACI,MAAM,MAAM,CAAN,I;aACL,IAAI,MAAM,CAAV,C;UACD,OAAO,MAAM,CAAN,I;;UAEP,OAAO,G;;MAEf,OAAO,EAAE,MAAM,CAAN,IAAF,K;K;IAGX,4E;MAe8E,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,c;MAC7G,WAAW,cAAX,EAAiB,SAAjB,EAA4B,OAA5B,C;MAEA,UAAU,S;MACV,WAAW,UAAU,CAAV,I;MAEX,OAAO,OAAO,IAAd,C;QACI,UAAW,GAAY,GAAN,IAAM,KAAK,C;QAC5B,aAAa,sBAAI,GAJJ,C;QACb,UAAU,UAAW,SAAQ,MAAR,EAAgB,OAAhB,C;QAErB,IAAI,MAAM,CAAV,C;UACI,MAAM,MAAM,CAAN,I;aACL,IAAI,MAAM,CAAV,C;UACD,OAAO,MAAM,CAAN,I;;UAEP,OAAO,G;;MAEf,OAAO,EAAE,MAAM,CAAN,IAAF,K;K;kGAGX,yB;MAAA,8D;MAAA,4D;MA5BqC,8D;QAAA,qB;UAAE,qBAAC,iBAAS,EAAT,CAAd,EAA4B,WAA5B,C;S;O;MAtBvC,+D;QAKBI,yB;UAAA,YAAiB,C;QACjB,uB;UAAA,UAAe,c;QAGf,+BAAa,SAAb,EAAwB,OAAxB,EAAiC,oCAAjC,C;O;KAtBJ,C;IA6BA,mE;MAmBoC,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,c;MACnE,WAAW,cAAX,EAAiB,SAAjB,EAA4B,OAA5B,C;MAEA,UAAU,S;MACV,WAAW,UAAU,CAAV,I;MAEX,OAAO,OAAO,IAAd,C;QACI,UAAW,GAAY,GAAN,IAAM,KAAK,C;QAC5B,aAAa,sBAAI,GAJJ,C;QACb,UAAU,UAAW,MAAX,C;QAEV,IAAI,MAAM,CAAV,C;UACI,MAAM,MAAM,CAAN,I;aACL,IAAI,MAAM,CAAV,C;UACD,OAAO,MAAM,CAAN,I;;UAEP,OAAO,G;;MAEf,OAAO,EAAE,MAAM,CAAN,IAAF,K;K;IAGX,8C;MAMQ,gBAAy,OA AZ,C;QAAuB,MAAM,gCAAYB,gBAAa,SAAb,mCAAKD,OAAID,OAAzB,C;WAC7B,gBAAy,CAAZ,C;QAAiB,MAAM,8BAA0B,gBAAa,SAAb,yBAA1B,C;WACvB,cAAU,IAAV,C;QAAkB,MAAM,8BAA0B,cAAW,OAAx,gCAA2C,IAA3C,OAA1B,C;K;IAchC,8B;MAEoC,MAAM,wBAAoB,8BAApB,C;K;IAE1C,8B;MAEoC,MAAM,wBAAoB,8BAApB,C;K;;;wF2Gjb1C,yB;M1GgCA,wE;M0GhCA,uC;QAmBW,kB1GqBiD,oB;Q0GM9C,Q;QAAA,OAAK,0B;QAAf,OAAU,cAAV,C;UAAU,mB;UACN,UAAU,sBAAM,CAAN,C;UACV,kBAaKB,sBAAY,GAAZ,C;UACIB,W1GuKJ,a0GvKgB,G1GuKhB,E0GrMyC,SA8BIB,CAAU,GAAV,EAAe,WAAf,EAA4B,CAA5B,EAA+B,uBAAuB,CAAC,WAAy,mBAAY,GAAZ,CAAnE,C1GuKvB,C;;Q0GrMA,OAGCO,W;O;KAnDX,C;4FAsBA,6C;MAwBc,Q;MAAA,OAAA,SAK,iB;MAAf,OAAU,cAAV,C;QAAU,mB;QACN,UAAU,sBAAM,CAAN,C;QACV,kBAaKB,sBAAY,GAAZ,C;QACIB,W1GuKJ,a0GvKgB,G1GuKhB,E0GvKuB,UAAU,GAAV,EAAe,WAAf,EAA4B,CAA5B,EAA+B,uBAAuB,CAAC,WAAy,mBAAY,GAAZ,CAAnE,C1GuKvB,C;;M0GrKA,OAAO,W;K;iFAGX,yB;MAAA,gB;MAAA,8B;M1GtBA,wE;M0GsBA,6D;QAnCW,kB1GqBiD,oB;Q0GM9C,Q;QAAA,OAAK,0B;QAAf,OAAU,cAAV,C;UAAU,mB;UACN,UAAU,sBAAM,CAAN,C;UACV,kBAaKB,sBAAY,GAAZ,C;U A8BwE,U;UA7B1F,W1GuKJ,a0GvKgB,G1GuKhB,E0G1IkC,UA7BD,GA6BC,EA7BoB,uBAAuB,CAAC,WAAy,mBAAY,GAAZ,CA6BzC,GAAW,qBA7B3B,GA6B2B,EA7BT,CA6BS,CAAX,GAA6C,UA7BxD,WA6BwD,6DAA5D,EA7BiB,CA6BjB,C1G0IIC,C;;Q0G3IA,OA1BO,W;O;KAGX,C;kFA0BA,yB;MAAA,gB;MAAA,8B;MAA A,0E;QAICc,Q;QAAA,OAAK,0B;QAAf,OAAU,cAAV,C;UAAU,mB;UACN,UAAU,sBAAM,CAAN,C;UACV,k BA6DQ,WA7DU,WAAy,GAAZ,C;UA6DuF,U;UAAjG,W1G2GZ,a0GvKgB,G1GuKhB,E0G3GiD,UA5DhB,GA4 DgB,EA5DK,uBAAuB,CA4DjE,WA5D8E,mBAAY,GAAZ,CA4D1B,GAAW,qBA5D1C,GA4D0C,EA5DxB,CA4 DwB,CAAX,GAA6C,UA5DvE,WA4DuE,6DAA5D,EA5DE,CA4DF,C1G2GjD,C;;Q0G5GA,OACY,W;O;KA7Bh B,C;iFAgCA,yB;MAAA,gB;MAAA,8B;M1GhFA,wE;M0GgFA,qD;QA7FW,kB1GqBiD,oB;Q0GM9C,Q;QAAA, OAAK,0B;QAAf,OAAU,cAAV,C;UAAU,mB;UACN,UAAU,sBAAM,CAAN,C;UACV,kBAaKB,sBAAY,GAAZ, C;UAKFiD,U;UAjFnE,W1GuKJ,a0GvKgB,G1GuKhB,E0GtFgC,UajFsB,uBAAuB,CAAC,WAAy,mBAAY,GAA Z,CAiFhD,kBAA6B,UajFjC,WaiFiC,6DAAvC,EajFmB,CAiFnB,C1GsFhC,C;;Q0GvFA,OA9EO,W;O;KA6DX, C;oFAoBA,yB;MAAA,gB;MAAA,8B;MAAA,kE;QatFc,Q;QAAA,OAAK,0B;QAAf,OAAU,cAAV,C;UAAU,mB; UACN,UAAU,sBAAM,CAAN,C;UACV,kBA2GQ,WA3GU,WAAy,GAAZ,C;UA2GgE,U;UAA1E,W1G6DZ,a0G vKgB,G1GuKhB,E0G7D+C,UA1GO,uBAAuB,CA0GjE,WA1G8E,mBAAY,GAAZ,CA0GjC,kBAA6B,UA1GhD, WA0GgD,6DAAvC,EA1GI,CA0GJ,C1G6D/C,C;;Q0G9DA,OACY,W;O;KAvBhB,C;qFA0BA,yB;MAAA,gB;MA AA,8B;M1G9HA,wE;M0G8HA,uC;QA3IW,kB1GqBiD,oB;Q0GM9C,Q;QAAA,OAAK,0B;QAAf,OAAU,cAA V,C;UAAU,mB;UACN,UAAU,sBAAM,CAAN,C;UACV,kBA6JQ,WA7JU,WAAy,GAAZ,C;UACC,oB;UA8Jc,U ;UAAjC,IA9JkD,uBAAuB,CA4JjE,WA5J8E,mBAAY,GAAZ,CA8JtF,C;YADA,mBA7J+C,C;;YA6J/C,mBACKB,

UA9JW,GA8JX,EAAe,UA9JC,WA8JD,6DAAf,EA9J6B,CA8J7B,C;;UAFV,W1GWZ,a0GvKGB,G1GuKhB,mB;;Q0GXA,OAAy,W;O;KAvBhB,C;IA6BA,6C;MARkC,Q;MAAA,OAAK,0B;MAAf,OAAU,cAAV,C;QAAU,mB;QACN,UAAU,sBAAM,CAAN,C;QACV,kBA+KG,WA/Ke,WAAy,GAAZ,C;QA2GgE,U;QAoE/E,W1GPP,a0GvKGB,G1GuKhB,E0GomC,CA9KmB,uBAAuB,CA8KtE,WA9KmF,mBAAY,GAAZ,CA0GjC,GAoErC,CAPeQc,GAA6B,UA1GhD,WA0GgD,6DAoEnD,IAAM,CAAN,I1GPnC,C;;M0GOA,OAAO,W;K;I+DnPOB,oC;MAAC,kB;MAAuB,kB;K;;wCAN7D,Y;MAMsC,iB;K;wCANtC,Y;MAM6D,iB;K;0CAN7D,wB;MAAA,wBAMsC,qCANtC,EAM6D,qCAN7D,C;K;sCAA,Y;MAAA,OAMsC,mDANtC,IAM6D,wCAN7D,O;K;sCAA,Y;MAAA,c;MAMsC,sD;MAAuB,sD;MAN7D,a;K;oCAAA,iB;MAAA,4IAMsC,sCANtC,IAM6D,sCAN7D,I;K;wFpKEA,yB;MAAA,kC;MAAAA,4C;MAAA,kD;QAMuF,wC;O;MANvF,4CAOI,Y;QAAuC,8B;O;MAP3C,8E;MAAA,2B;QAMuF,2C;O;KANvF,C;IAcsC,2C;MAAC,wC;K;0CACnC,Y;MAAqD,4BAAiB,wBAAjB,C;K;;IAIzD,yC;MAI4D,OAAI,oCAAJ,GAA2B,SAAK,KAAhC,GAA0C,I;K;IAEtG,uD;MAIOE,OAAI,oCAAJ,GAA2B,SAAK,KAAhC,GAA0C,S;K;IAGpH,8B;MAMoB,Q;MADhB,aAAa,gB;MACG,2B;MAAhB,OAAgB,cAAhB,C;QAAgB,yB;QACL,OAAP,MAAO,EAAO,OAAP,C;;MAEX,OAAO,M;K;IAGX,4B;MAUiB,Q;MAHb,mBAAmB,mCAAwB,EAAXB,C;MACnB,YAAy,iBAaA,YAAb,C;MACZ,YAAy,iBAaA,YAAb,C;MACC,2B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,KAAM,WAAI,IAAK,MAAT,C;QACN,KAAM,WAAI,IAAK,OAAT,C;;MAEV,OAAO,UAAS,KAAT,C;K;wFUxDX,qB;MAKqE,gB;K;IAErE,iC;MAMoE,4BAAiB,SAAjB,C;K;uFAEpE,gC;MAKI,OAAgB,mBAAhB,C;QAAgB,8B;QAAM,UAAU,OAAV,C;;K;IAMY,oC;MAAC,0B;MACnC,eAAoB,C;K;yCACpB,Y;MAAwC,OAAA,eAAS,U;K;sCACjD,Y;MAA6E,Q;MAAhC,wBAaA,oBAAmB,mBAAnB,EAAMB,2BAAnB,QAAb,EAA0C,eAAS,OAAAnD,C;K;;sF2J5BjD,yB;MAAA,4E;MAAA,gB;MAAA,8B;MAAA,+C;QAUiC,Q;QAA7B,OAA6B,wCAAqB,QAAS,aAA9B,0D;O;KAVjC,C;wFAYA,yB;MAAA,4E;MAAA,gB;MAAA,8B;MAAA,+C;QAWiC,Q;QAA7B,OAA6B,wCAAqB,QAAS,aAA9B,0D;O;KAXjC,C;sFAaA,+C;MAQI,SAAK,aAAI,QAAS,aAAb,EAAMB,KAAAnB,C;K;ICnCT,8C;MAUI,IAAI,wCAAJ,C;QACI,OAAO,SAAK,4BAAqB,GAARb,C;MAET,4B;M3KoTI,Q;MALX,YAAy,oB2K/Sa,G3K+Sb,C;MACZ,IAAI,iBAAiB,CAAC,4B2KhTG,G3KgTH,CAAtB,C;Q2KhTgC,MAAM,2BAAuB,wCAAvB,C;;Q3KoTIC,2BAAO,sE;;M2KpTX,+B;K;IAGJ,8C;MAUQ,kBADE,SACF,kB;QADJ,OACkC,YAAT,SAAK,IAAI,EAAY,YAAZ,C;;QADIC,OAey,uBAAMB,SAAnB,EAAYB,YAAzB,C;K;IAGhB,gD;MAWQ,kBADE,SACF,yB;QADJ,OACyC,cAAT,SAAK,IAAI,EAAY,YAAZ,C;;QADzC,OAey,8BAA0B,SAAI1B,EAAGC,YAAhC,C;K;;;IAc0B,4C;MAAC,wB;MAAoC,0B;K;qEAApC,Y;MAAA,yB;K;0CACvC,iB;MAA4C,OAAI,OAAJ,QAAI,EAAO,KAAP,C;K;4CAChD,Y;MAA+B,OAAI,SAAJ,QAAI,C;K;4CACnC,Y;MAAkC,OAAA,QAAI,W;K;0FACf,Y;MAAQ,OAAA,QAAI,K;K;2CACnC,Y;MAAkC,OAAA,QAAI,U;K;qDACtC,e;MAA4C,OAAA,QAAI,mBAAY,GAAZ,C;K;uDACHD,iB;MAAgE,OAAA,QAAI,qBAAC,KAAAd,C;K;6CACpE,e;MAA+B,OAAA,QAAI,WAAI,GAAJ,C;K;0FACT,Y;MAAQ,OAAA,QAAI,K;K;4FACH,Y;MAAQ,OAAA,QAAI,O;K;6FACJ,Y;MAAQ,OAAA,QAAI,Q;K;8DAEvD,e;MAAmD,gBAAJ,Q;MAAI,4B;M3K4PxC,Q;MALX,YAAy,oB2KvPyD,G3KuPzD,C;MACZ,IAAI,iBAAiB,CAAC,4B2KxP+C,G3KwP/C,CAAtB,C;QACI,2B2KzPwE,mB;;Q3K4PxE,2BAAO,sE;;M2K5PoC,+B;K;;IAGN,mD;MAAC,wB;MAA2C,0B;K;4EAA3C,Y;MAAA,yB;K;iDAC1C,iB;MAA4C,OAAI,OAAJ,QAAI,EAAO,KAAP,C;K;mDACHD,Y;MAA+B,OAAI,SAAJ,QAAI,C;K;mDACnC,Y;MAAkC,OAAA,QAAI,W;K;iGACf,Y;MAAQ,OAAA,QAAI,K;K;kDACnC,Y;MAAkC,OAAA,QAAI,U;K;4DACtC,e;MAA4C,OAAA,QAAI,mBAAY,GAAZ,C;K;8DACHD,iB;MAAgE,OAAA,QAAI,qBAAC,KAAAd,C;K;oDACpE,e;MAA+B,OAAA,QAAI,WAAI,GAAJ,C;K;iGACf,Y;MAAQ,OAAA,QAAI,K;K;mGACH,Y;MAAQ,OAAA,QAAI,O;K;oGACU,Y;MAAQ,OAAA,QAAI,Q;K;sDAE5E,sB;MAAyC,OAAA,QAAI,aAAI,GAAJ,EAAS,KAAT,C;K;uDAC7C,e;MAAkC,OAAA,QAAI,cAAO,GAAP,C;K;yDACtC,gB;MAA2C,QAAI,gBAAO,IAAP,C;K;gDAC/C,Y;MAAuB,QAAI,Q;K;qEAE3B,e;MAAmD,gBAAJ,Q;MAAI,4B;M3KuOxC,Q;MALX,YAAy,oB2KIOyD,G3KkOzD,C;MACZ,IAAI,iBAAiB,CAAC,4B2KnO+C,G3KmO/C,CAAtB,C;QACI,2B2KpOwE,mB;;Q3KuOxE,2BAAO,sE;;M2KvOoC,+B;K;;I3KvFnD,oB;MAAA,wB;MACI,8C;K;gCAEA,iB;MAA4C,oCAAsB,KAAM,U;K;kCACxE,Y;MAA+B,Q;K;kCAC/B,Y;MAAkC,W;K;gFAEX,Y;MAAQ,Q;K;iCAC/B,Y;MAAkC,W;K;2CAEIC,e;MAA+C,Y;K;6CAC/C,iB;MAAsD,Y;K;mCACtD,e;MAAwC,W;K;mFACY,Y;MAAQ,6B;K;gFAC/B,Y;MAAQ,6B;K;kFACI,Y;MAAQ,8B;K;uCAEjD,Y;MAAiC,6B;K;;IAjBrC,gC;MAAA,+B;QAAA,c;OAAA,wB;K;IAoBa,oB;MAMuE,Q;MAA7B,OAA6B,uE;K;IAEvE,wB;MAAIL,OAAI,KAAM,OAAN,GAAa,CAAjB,GAA0B,QAAAN,KAAM,EAAM,qBAAC,YAAy,KAAM,OAAIB,CAAd,CAAN,CAA1B,GAA6E,U;K;kFAEjF,yB;MAAA,oD;MAAA,mB;QAO8C,iB;O;KAP9C,C;8FASA,yB;MAAA,wE;M

AAA,mB;QAQ4D,2B;O;KAR5D,C;IAUA,+B;MAYiD,gBAA7C,qBAAoB,YAAy,KAAM,OAAIB,CAApB,C;MA  
AqD,wB;MAArD,OUJO,S;K;wFVMX,yB;MAAA,4D;MAAA,mB;QAOSD,qB;O;KAPtD,C;IASA,4B;MAM8G,gB  
AAvC,eAAc,YAAy,KAAM,OAAIB,CAAd,C;MAA+C,wB;MAA/C,OUrB5D,S;K;4FVvBX,yB;MAAA,wE;MAA  
A,mB;QAK8D,2B;O;KAL9D,C;IAOA,8B;MAU+E,OAAM,QAAN,KAAM,EAAM,qBAAc,YAAy,KAAM,OAAI  
B,CAAd,CAAN,C;K;sFAErF,yB;MchBA,wE;MdgBA,gC;QcZiC,gBAAtB,oB;Qd8BiB,aU7DxB,W;QV6DA,OU5  
DO,SI8B2C,Q;O;KdYtD,C;uFA0BA,yB;McNCA,uE;MdmCA,0C;Qc/ByC,gBAA9B,mBdqDiB,QcrDjB,C;QdqD2B  
,aU3FIC,W;QV2FA,OU1FO,SIqCmD,Q;O;Kd+B9D,C;4FAoCA,qB;MAK+D,QAAC,mB;K;kGAehE,qB;MAWI,O  
AAO,qBAAgB,mB;K;sFAG3B,yB;MAAA,oD;MAAA,4B;QAM2D,uCAAQ,U;O;KANnE,C;sFAQA,mC;MASI,O  
AAI,mBAAJ,GAAe,cAAf,GAAMC,S;K;yFAEvC,yB;MAyBA,kC;MAAA,8B;MAzBA,iC;QAgiC,Q;QAx2E,O  
AwBxD,CAAnB,wDAAmB,oBAXBoE,GAwBpE,C;O;KAhCpD,C;+EAUA,yB;MAAA,kC;MAAA,8B;MAAA,iC;  
QAKiC,Q;QAA7B,OAGd,CAAnB,wDAAmB,YAAI,GAAJ,C;O;KALpD,C;+EAOA,iC;MAKI,sBAAI,GAJ,EA  
AS,KAAT,C;K;4FAGJ,yB;MAAA,kC;MAAA,8B;MAAA,iC;QAOiC,Q;QAA7B,OAGd,CAAnB,wDAAmB,oBA  
AY,GAAZ,C;O;KAPpD,C;gGASA,4B;MASsG,OAAA,SAAK,qBAAc,KAAd,C;K;kFAG3G,yB;MAAA,gD;MAA  
A,8B;MAAA,iC;QASiC,Q;QAA7B,OAAuD,CAA1B,+DAA0B,eAAO,GAAP,C;O;KAT3D,C;6FAWA,qB;MAWo  
E,oB;K;6FAEpE,qB;MAWoE,sB;K;kFAEpE,yB;MAAA,6B;MAAA,4B;QAIgE,qBAAK,aAAL,EAAU,eAAV,C;O;  
KAJhE,C;2FAMA,wC;MAMiF,Q;MAAA,mCAAI,GAJ,oBAAy,c;K;uGAG7F,yB;MAAA,gB;MAAA,8B;MAA  
A,+C;QAMe,Q;QALX,YAAy,oBAAI,GAJ,C;QACZ,IAAI,iBAAiB,CAAC,4BAAy,GAAZ,CAAtB,C;UACI,OA  
AO,c;;UAGP,OAAO,sE;;O;KANf,C;IAUA,oC;MAUkD,uCAAqB,GAARb,C;K;sFAEID,wC;MAUW,Q;MADP,YA  
AY,oBAAI,GAJ,C;MACL,IAAI,aAAJ,C;QACH,aAAa,c;QACb,sBAAI,GAJ,EAAS,MAAT,C;QACA,a;;QAEA  
,Y;;MALJ,W;K;wFASJ,qB;MAMwF,OAAA,iBAAQ,W;K;wFAEHg,qB;MAMgH,OAAA,iBAAQ,W;K;4FAExH,6  
C;Maq1BoB,Q;MAAA,Obh1BT,iBag1BS,W;MAAhB,OAGb,cAAhB,C;QAAGb,yB;Qbh1Ba,Wai1Bb,aAAGb,O  
bj1Be,Iai1B/B,Ebj1BsC,Sai1BZ,CAAE,OAAf,CAA1B,C;;Mbj1BhB,OAA6B,W;K;wFAGjC,6C;Ma60BoB,Q;MAA  
A,Obr0BT,iBaq0BS,W;MAAhB,OAGb,cAAhB,C;QAAGb,yB;Qbr0Ba,Was0Bb,abt0B0B,Sas0BtB,CAAY,OAAZ  
,CAAJ,EAAYC,Obt0BC,Mas0B1C,C;;Mbt0BhB,OAA6B,W;K;IAGjC,kC;MAIyB,Q;MAArB,wBAAqB,KAARb,g  
B;QAAqB,aAAA,KAARb,M;QAAK,IAAC,yBAAD,EAAM,2B;QACP,sBAAI,GAJ,EAAS,KAAT,C;;K;IAIR,oC;  
MAIyB,Q;MAAA,uB;MAArB,OAAqB,cAARb,C;QAAqB,wB;QAAhB,IAAC,yBAAD,EAAM,2B;QACP,sBAAI,  
GAJ,EAAS,KAAT,C;;K;IAIR,oC;MAIyB,Q;MAAA,uB;MAArB,OAAqB,cAARb,C;QAAqB,wB;QAAhB,IAAC,  
yBAAD,EAAM,2B;QACP,sBAAI,GAJ,EAAS,KAAT,C;;K;wFAIR,yB;MAAA,0D;MAAA,uE;MAAA,uC;QAS  
W,kBAAy,mBAAoB,YAAy,CAAZ,CAApB,C;Qa8xBH,Q;QAAA,Obh1BT,iBag1BS,W;QAAhB,OAGb,cAAhB,  
C;UAGb,yB;Ubh1Ba,Wai1Bb,aAAGb,Obj1Be,Iai1B/B,Eb/xB2C,Sa+xBjB,CAAE,OAAf,CAA1B,C;;Qb/xBhB,O  
AID6B,W;O;KAYCjC,C;oFAYa,yB;MAAA,0D;MAAA,uE;MAAA,uC;QAYW,kBAAU,mBAAoB,YAAy,CAAZ,  
CAApB,C;Qa+wBD,Q;QAAA,Obr0BT,iBaq0BS,W;QAAhB,OAGb,cAAhB,C;UAGb,yB;Ubr0Ba,Was0Bb,abh  
xByC,SagxBrc,CAAY,OAAZ,CAAJ,EAAYC,Obt0BC,Mas0B1C,C;;QbhxBhB,OAtD6B,W;O;KA0CjC,C;0FAeA,y  
B;MAAA,wE;MAAA,uC;QAQkB,Q;QADd,aAAa,oB;QACC,OAAA,SA3FsE,QAAQ,W;QA2F5F,OAAc,cAAd,C;  
UAAc,uB;UACV,IAAI,UAAU,KAAM,IAAhB,CAAJ,C;YACI,MAAO,aAAI,KAAM,IAAV,EAae,KAAM,MAAr  
B,C;;QAGf,OAAO,M;O;KAbX,C;8FAGbA,yB;MAAA,wE;MAAA,uC;QAQkB,Q;QADd,aAAa,oB;QACC,OAAA,  
SA3GsE,QAAQ,W;QA2G5F,OAAc,cAAd,C;UAAc,uB;UACV,IAAI,UAAU,KAAM,MAAhB,CAAJ,C;YACI,MA  
AO,aAAI,KAAM,IAAV,EAae,KAAM,MAArB,C;;QAGf,OAAO,M;O;KAbX,C;yFAiBA,6C;MAOoB,Q;MAAA,  
OAAA,SA3HoE,QAAQ,W;MA2H5F,OAGb,cAAhB,C;QAAGb,yB;QACZ,IAAI,UAAU,OAAV,CAAJ,C;UACI,  
WAAY,aAAI,OAAQ,IAAZ,EAAiB,OAAQ,MAAZB,C;;MAGpB,OAAO,W;K;qFAGX,yB;MAAA,wE;MAAA,uC;  
QAOW,kBAAS,oB;QAFa,Q;QAAA,OA3HoE,iBAAQ,W;QA2H5F,OAGb,cAAhB,C;UAGb,yB;UACZ,IAcmC,  
SAd/B,CAAU,OAAV,CAAJ,C;YACI,WAAY,aAAI,OAAQ,IAAZ,EAAiB,OAAQ,MAAZB,C;;QAapB,OAVO,W;  
O;KAGX,C;+FAUA,6C;MAOoB,Q;MAAA,OAAA,SAPJoE,QAAQ,W;MAoJ5F,OAGb,cAAhB,C;QAAGb,yB;Q  
ACZ,IAAI,CAAC,UAAU,OAAV,CAAL,C;UACI,WAAY,aAAI,OAAQ,IAAZ,EAAiB,OAAQ,MAAZB,C;;MAGp  
B,OAAO,W;K;2FAGX,yB;MAAA,wE;MAAA,uC;QAOW,kBAAy,oB;QAFh,Q;QAAA,OAPJoE,iBAAQ,W;QAoJ  
5F,OAGb,cAAhB,C;UAGb,yB;UACZ,IAAI,CACKC,SADjC,CAAU,OAAV,CAAL,C;YACI,WAAY,aAAI,OAA  
Q,IAAZ,EAAiB,OAAQ,MAAZB,C;;QAapB,OAVO,W;O;KAGX,C;IAUA,0B;MAQqB,IAAN,I;MADX,IAAI,oCA  
AJ,C;QACW,QAAM,cAAN,C;eACH,C;YAAK,iB;YAAL,K;eACA,C;YAAK,aAAU,8BAAJ,GAAKB,sBAAK,CA

AL,CAAIB,GAA+B,oBAAW,OAAhD,C;YAAL,K;kBACQ,0BAAM,qBAAoB,YAAY,cAAZ,CAApB,CAAN,C;Y  
AHL,K;;QAAP,W;OAMJ,OAAoC,oBAA7B,mBAAM,oBAAN,CAA6B,C;K;IAGxC,yC;MAIwB,SAApB,WAAoB  
,Y;MAApB,kB;K;IAEJ,4B;MAM6D,QAAM,gBAAN,C;aACzD,C;UADyD,OACpD,U;aACL,C;UAFyD,OAEPD,  
MAAM,UAAK,CAAL,CAAN,C;gBAFoD,OAGjD,mBAAM,qBAAoB,YAAY,gBAAZ,CAApB,CAAN,C;;K;IAG  
Z,yC;MAIwB,OAApB,WAAoB,Y;MAApB,kB;K;IAEJ,4B;MAM4D,OAA6B,oBAA7B,mBAAM,oBAAN,CAA6B  
,C;K;IAEzF,yC;MAIwB,SAApB,WAAoB,Y;MAApB,kB;K;IAEJ,4B;MAMqD,QAAM,cAAN,C;aACjD,C;UADiD  
,OAC5C,U;aACL,C;UAFiD,OC/X8B,uB;gBd+X9B,OAGzC,uB;;K;IAGZ,iC;MAMmE,4BAAc,SAAd,C;K;IAEnE,  
yC;MAKI,WAAoB,0B;MAApB,kB;K;IAEJ,kC;MAOI,Q;MAAA,IAAI,SAAK,UAAT,C;QAAA,OAAoB,MAAM,I  
AAN,C;;QAAqC,kBAApB,qBAAc,SAAd,C;QAA4B,wBAAS,UAAT,EAAqB,WAArB,C;QAAjE,OUhiBO,W;;M  
VgiBP,W;K;IAEJ,mC;MAOI,Q;MAAA,IAAI,SAAK,UAAT,C;QAAA,OAA0B,MAAN,KAAM,C;;QAAiC,kBAAp  
B,qBAAc,SAAd,C;QAA4B,4B;QAAnE,OUziBO,W;;MVyiBP,W;K;IAEJ,mC;MAOI,Q;MAAA,IAAI,SAAK,UA  
T,C;QAAA,OAA0B,QAAN,KAAM,C;;QAAiC,kBAApB,qBAAc,SAAd,C;QAA4B,0B;QAAnE,OUljBO,W;;MVkj  
BP,W;K;IAEJ,mC;MAOwB,kBAApB,qBAAc,SAAd,C;MAA4B,4B;MAA5B,OAA4C,oBU3jBrC,WV2jBqC,C;K;I  
AEhD,iC;MAOwB,kBAApB,qBAAc,SAAd,C;MAA4B,+B;MAA5B,OUpkBO,W;K;0FVukBX,2B;MAKI,sBAAI,I  
AAK,MAAT,EAAgB,IAAK,OAArB,C;K;4FAGJ,yB;MAAA,gD;MAAA,mC;QAKI,kBAAO,KAAP,C;O;KALJ,C;  
4FAQA,yB;MAAA,gD;MAAA,mC;QAKI,kBAAO,KAAP,C;O;KALJ,C;4FAQA,yB;MAAA,gD;MAAA,mC;QAK  
I,kBAAO,KAAP,C;O;KALJ,C;4FAQA,0B;MAKI,yBAAO,GAAP,C;K;IAGJ,kC;MAOwB,kBAAf,aAAL,SAAK,C;  
MAsCL,6B;MAtCA,OAA+C,oBUtnBxC,WVsnBwC,C;K;IAEnD,mC;MAQwB,kBAAf,aAAL,SAAK,C;MAqCK,  
YAAL,gBAAK,O;MArCV,OAAgD,oBUhoBzC,WVgoByC,C;K;IAEpD,mC;MAQwB,kBAAf,aAAL,SAAK,C;MA  
oCK,YAAL,gBAAK,O;MApCV,OAAgD,oBU1oBzC,WV0oByC,C;K;IAEpD,mC;MAQwB,kBAAf,aAAL,SAAK,  
C;MAmCK,YAAL,gBAAK,O;MAncV,OAAgD,oBUppBzC,WVopByC,C;K;4FAEpD,0B;MAMI,uBAAO,GAAP,  
C;K;8FAGJ,yB;MAAA,sD;MAAA,kC;QAMc,UAAV,SAAK,KAAP,EAAU,IAAV,C;O;KANd,C;8FASA,yB;MA  
AA,sD;MAAA,kC;QAMc,UAAV,SAAK,KAAP,EAAU,IAAV,C;O;KANd,C;8FASA,yB;MAAA,sD;MAAA,kC;Q  
AMc,UAAV,SAAK,KAAP,EAAU,IAAV,C;O;KANd,C;IAUA,wC;MACsD,QAAM,cAAN,C;aACID,C;UADkD,O  
AC7C,U;aACL,C;UAFkD,gB;gBAAA,OAG1C,S;;K;oF4KtwBZ,yB;MAAA,8D;MAAA,8B;MAAA,qC;QAUiC,Q;  
QAA7B,OAA2D,CAA9B,sEAA8B,eAAO,OAAP,C;O;KAV/D,C;wFAYA,yB;MAAA,8D;MAAA,8B;MAAA,sC;  
QASiC,Q;QAA7B,OAA2D,CAA9B,sEAA8B,oBAAU,QAAP,C;O;KAT/D,C;wFAWA,yB;MAAA,8D;MAAA,8B;  
MAAA,sC;QASiC,Q;QAA7B,OAA2D,CAA9B,sEAA8B,oBAAU,QAAP,C;O;KAT/D,C;4FAWA,8B;MAKI,SA  
K,WAAI,OAAJ,C;K;4FAGT,yB;MAAA,gD;MAAA,sC;QAKS,OAAL,SAAK,EAAO,QAAP,C;O;KALT,C;4FAQ  
A,yB;MAAA,gD;MAAA,sC;QAKS,OAAL,SAAK,EAAO,QAAP,C;O;KALT,C;4FAQA,yB;MAAA,gD;MAAA,sC  
;QAKS,OAAL,SAAK,EAAO,QAAP,C;O;KALT,C;8FAQA,8B;MAKI,SAAK,cAAO,OAAP,C;K;8FAGT,yB;MA  
A,sD;MAAA,sC;QAKS,UAAL,SAAK,EAAU,QAAP,C;O;KALT,C;8FAQA,yB;MAAA,sD;MAAA,sC;QAKS,UA  
AL,SAAK,EAAU,QAAP,C;O;KALT,C;8FAQA,yB;MAAA,sD;MAAA,sC;QAKS,UAAL,SAAK,EAAU,QAAP,C  
;O;KALT,C;IAQA,qC;MAIU,IAIe,I;MAHjB,kBADE,QACF,c;QAAiB,OAAO,yBAAO,QAAP,C;;QAEpB,aAAsB,  
K;QACT,0B;QAAb,OAAa,cAAb,C;UAAa,sB;UACT,IAAI,oBAAI,IAAJ,CAAJ,C;YAAe,SAAS,I;;QAC5B,OAAO  
,M;;K;IAKnB,uC;MAKiB,Q;MADb,aAAsB,K;MACT,0B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,IAAI,oBAAI,I  
AAJ,CAAJ,C;UAAe,SAAS,I;;MAE5B,OAAO,M;K;IAGX,uC;MAII,OAAO,yBAAgB,OAAT,QAAS,CAAhB,C;K;  
IAGX,0C;MAIW,iBAAmB,gCAAT,QAAS,EAAgC,SAAhC,C;MAIHG,Q;MAKH7B,OAIH2D,CAA9B,sEAA8B,o  
BAAU,UAAV,C;K;IAqH/D,0C;MAII,UAAmB,8BAAT,QAAS,C;MACnB,O7K0EwD,C6K1EjD,G7K0EkD,U6K1  
EID,IAAoB,4BAAU,GAAV,C;K;IAG/B,0C;MAII,OnLqoPO,EmLroPA,QnL6jPA,YAAQ,CAwER,CmLroPA,IAA  
yB,4BAAmB,8BAAT,QAAS,CAAnB,C;K;IAGpC,0C;MAIW,iBAAmB,gCAAT,QAAS,EAAgC,SAAhC,C;MA7H  
G,Q;MA6H7B,OA7H2D,CAA9B,sEAA8B,oBAAU,UAAV,C;K;IAGl/D,0C;MAII,InLunPO,EmLvnPH,QnL+iPG,  
YAAQ,CAwER,CmLvnPP,C;QACI,OAAO,4BAAmB,8BAAT,QAAS,CAAnB,C;;QAEp,OAAO,wB;K;IAGf,0C;  
MAII,UAAmB,8BAAT,QAAS,C;MACnB,I7K0CwD,C6K1CpD,G7K0CqD,U6K1CzD,C;QACI,OAAO,4BAAU,G  
AAV,C;;QAEp,OAAO,wB;K;IAGf,kC;MACI,a7KmCwD,CAAC,mB;M6K1CzD,iB;MACA,OAAO,M;K;IAIX,2C;  
MAKkF,gCAAc,SAAd,EAAyB,IAAZB,C;K;IAEIF,2C;MAKkF,gCAAc,SAAd,EAAyB,KAABZB,C;K;IAEIF,sE;MA  
CI,iBAAa,KAAb,C;MIKIJgB,kBkKmJX,oB;MACD,OAAO,qBAAP,C;QACI,IAAI,UAAU,kBAAV,6BAAJ,C;UA  
CI,oB;UACA,WAAS,I;SAGrB,OAAO,Q;K;oFAIX,4B;MAM6D,kCAAS,KAAT,C;K;IAE7D,gC;MAKiD,IAAI,m

BAAJ,C;QA Ae,MAAM,2BA AuB,gBA AvB,C;;QA ArB,OA AmE,2BA AS,CA AT,C;K;IA EpH,sC;MA KwD,OA AI,mBA AJ,GAA e,IA Af,GAA yB,2BA AS,CA AT,C;K;IA EjF,+B;MA KgD,IA AI,mBA AJ,C;QA Ae,MA AM,2BA AuB,gBA AvB,C;;QA ArB,OA AmE,2BA AS,2BA AT,C;K;IA EnH,qC;MA KuD,OA AI,mBA AJ,GAA e,IA Af,GAA yB,2BA AS,2BA AT,C;K;IA EhF,2C;MA K8E,kCA Ac,SA Ad,EA AyB,IA AzB,C;K;IA E9E,2C;MA K8E,kCA Ac,SA Ad,EA AyB,KAA zB,C;K;IA E9E,wE;MA EgB,UAGS,MA HT,EA cY,MA dZ,EA c6B,M;MA fzC,IA AI,uCA AJ,C;QA CI,OA AoC,cAA 5B,sEA A4B,EA Ac,SA Ad,EA AyB,uBA AzB,C;MA ExC,iBA AsB,C;MA CD,oC;MA ArB,qBA AkB,CA AI B,mC;QA CI,cAA c,sBA AK,SA AL,C;QA Cd,IA AI,UAA U,OA AV,MA AsB,uBA A1B,C;UA CI,Q;QA EJ,IA AI,eAA c,SA AI B,C;UA CI,sBA AK,UA AL,EA AmB,OA AnB,C;QA EJ,+B;;MA EJ,IA AI,aAA a,cAA jB,C;QA CwB,oC;QA AiB,mB;QA ArC,oE;UA CI,2BA AS,WA AT,C;QA EJ,OAA O,I;;QA EP,OAA O,K;;K;IChS+B,wC;MA AkC,uB;MA AjC,OB;K;4FACpB,Y;MA AQ,OAAA,eA AS,K;K;iDACx C,iB;MA AkC,mCA AS,OBAA oB,KAA pB,CA AT,C;K;;IAGT,gC;M AA yC,8B;MA Ax C,OB;K;oFACH,Y;MA AQ,OAAA,eA AS,K;K;yCACx C,iB;MA AkC,mCA AS,OBAA oB,KAA pB,CA AT,C;K;mCAEIC,Y;MA AuB,eA AS,Q;K;8CAC hC,iB;MA AuC,OAAA,eA AS,kBA AS,OBAA oB,KAA pB,CA AT,C;K;yCA EhD,OB;MA A8C,OAAA,eA AS,aAA I,OBAA oB,KAA pB,CA AJ,EA AgC,OAA hC,C;K;yCACvD,OB;MA CI,eA AS,aAA I,2BA AqB,KAA rB,CA AJ,EA AiC,OAA jC,C;K;;IA IjB,+C;MA CoB,Q;MA AA,kC;MA AhB,IA Aa,CA AT,OBAA J,C;QAAA,OAA 2B,8BA AY,KAA Z,I;;QA AuB,MA AM,8BA A0B,mBA AgB,KAA hB,2BA A0C,gBA AG,2BA AH,CAA 1C,OAA 1B,C;K;IA E5D,gD;MA CoB,Q;MA AA,qB;MA hB,IA Aa,CA AT,OBAA J,C;QAAA,OAA sB,iBA AO,KAA P,I;;QA AkB,MA AM,8BA A0B,oBA AiB,KAA jB,2BA A2C,gBA AG,cAA H,CAA 3C,OAA 1B,C;K;IAG ID,+B;MA K+C,gCAA qB,SA ArB,C;K;IA E/C,iC;MAM6D,wBA Aa,SA Ab,C;K;;;IvKpC7D,oD;MA QuF,wC;K;IAR vF,8CASI,Y;MA AuC,8B;K;IAT3C,gF;IwKY8G,wC;MA AA,mB;QA AE,kBA AS,aA AT,C;O;K;IAThH,yB;MASqG,oCA AS,sBA AT,C;K;8FAErG,yB;MA AA,kD;MxKdA,kC;MA AA,0C;MA AA,kD;QA QuF,wC;O;MARvF,4CASI,Y;QA AuC,8B;O;MAT3C,8E;MwKiB2I,qD;QAAA,mB;UAA E,gBA AS,qBA AT,C;S;O;MAH7I,gC;QAGkI,kCA AS,mCA AT,C;O;KAHII,C;IA KA,2B;MA QI,eAA e,6B;MA Cf,oBA A0B,+BA AN,KAA M,EA AwC,QA Ax C,EA A+D,Q AA/D,C;MA C1B,OAA O,Q;K;8FAGX,yB;MA AA,kD;MA AA,gC;QAGkI,gBA AS,aA AT,C;O;KAHII,C;IAGB0C,y B;K;+CA oBtC,kC;MA OI,IA AI,uCAA 0B,QA AS,UAA vC,C;QA AkD,M;MA CID,OAA O,sBA AS,QA AS,WA AI B,e;K;+CAGX,kC;MA QqD,6BA AS,QA AS,WA AI B,e;K;;;;;IA ezD,mC;MA A2C,wB;MA CvC,eAA oB,C;MA CpB,mB AA4B,I;MA C5B,sBA AyC,I;MA CzC,gBA AoC,I;K;gDAEpC,Y;MA CI,OAA O,IA AP,C;QA CI,QA AM,YA AN,C;eA CI,C;YAAA,K;eACA,C;YACI,IA AI,kCA Ae,UA AnB,C;cACI,eAA Q,C;cACR,OAA O,I;;cAEP,sBA Ae,I;;YALvB,K;eAOA,C;YAAc,OAA O,K;eACrB,C;eAAA,C;YAAgC,OAA O,I;kBAC/B,MA AM,yB;;QAGIB,eAA Q,C;QACR,WA AW,4B;QACX,gBA AW,I;QACX,IxH/FR,oBDgDQ,WyH+CY,kBzH/CZ,CChDR,C;;K;6CwHmGA,Y;MACU,IA Se,I;MATrB,QA AM,YA AN,C;aACI,C;aAAA,C;UA AsC,OAA O,qB;aAC7C,C;UA CI,eAA Q,C;UACR,OAA O,kCAA e,O;aAE1B,C;UA CI,eAA Q,C;UACR,aACa,mF;UACb,mBA AY,I;UACZ,OAA O,M;gBAEH,MA AM,yB;;K;uDAItB,Y;MA CI,IA AI,CA AC,cAAL,C;QA AgB,MA AM,6B;;QAA8B,OAA O,W;K;2DAG/D,Y;MA A4C,QA AM,YA AN,C;aACx C,C;UADwC,OAC1B,6B;aACd,C;UAFwC,OA ExB,6BA AsB,sBA AtB,C;gBAFwB,OAGhC,6BA AsB,uCAA oC,YAA 1D,C;;K;IA OqC,4E;MA AA,oB;QACzC,wCA AW,C;QA AX,OACA,yB;O;K;oDALR,+B;MA CI,mBA AY,K;MA CZ,eAA Q,C;MA CR,OAA 6C,0CA AtC,c;K;IA UsC,+E;MA AA,oB;QACzC,wCA AW,C;QA AX,OACA,yB;O;K;yDANR,kC;MA CI,IA AI,CA AC,QA AS,UAA d,C;QA AyB,M;MA CzB,sBA Ae,Q;MA Cf,eAA Q,C;MA CR,OAA 6C,6CA AtC,c;K;2DAMX,kB;MzHjBO,Q;MADP,eyHoBI,MzHpBJ,C;MACO,QyHmBH,MzHnBG,+D;MyHoBH,eAA Q,C;K;kGAIR,Y;MA AQ,0C;K;;IxK1LhB,oD;MA QuF,wC;K;IARvF,8CASI,Y;MA AuC,8B;K;IAT3C,gF;sFAAA,yB;MA AA,kC;MA AA,0C;MA AA,kD;QA QuF,wC;O;MARvF,4CASI,Y;QA AuC,8B;O;MAT3C,8E;MA AA,2B;QA QuF,2C;O;KARvF,C;IA iBgE,+C;MA AA,mB;QA AE,sB;O;K;IALIE,kC;MA KuD,OA AkB,2CA AT,+BA AS,E;K;IA EzE,8B;MA K6D,OA AI,Qb2rPtD,YAA Q,Ca3rP0C,GAA wB,eAA xB,GA AsD,WA AT,QA AS,C;K;IA EnH,yB;MAG8C,kC;K;IA E9C,yB;MA AA,6B;K;uCACI,Y;MA A6C,kC;K;2CAC7C,a;MA A4B,kC;K;2CAC5B,a;MA A4B,kC;K;;IA HhC,qC;MA AA,oC;QAAA,mB;OAAA,6B;K;oFAMA,yB;MA AA,2D;MA AA,4B;QAM4D,uCAA Q,e;O;KANpE,C;IAGB4F,mH;MA AA,wC;MA AA,6B;MA AA,yB;MA AA,wC;MA AA,wD;MA AA,kC;K;;;kDAAA,Y;;;;cACxF,eAA e,uBA Aa,W;cAC5B,IA AI,QA AS,UAA b,C;gBACI,gB;gCAAA,sCA AS,QA AT,O;oBAAA,2C;yBAAA,yB;gBAAA,Q;;gBAEA,gB;gCAAA,sCA AS,iCA AT,O;oBAAA,2C;yBAAA,yB;gBAAA,Q;;;;cAJJ,W;;cAAA,W;;;;;K;IADwF,gE;MA AA,yD;uBAAA,uG;YAAA,S;iBAAA,Q;;iBAAA,uB;O;K;IAP5F,4C;MA OmF,gBA AS,uCA AT,C;K;IAGBb,4B;MA AE,OAAA,EA AG,W;K;IAP3E,8B;MA O8D,4BA AQ,cA AR,C;K;IAUQ,8B;MA

AE,OAAA,EAAG,W;K;IAR3E,8B;MAQ8D,4BAAQ,gBAAR,C;K;IAM1B,8B;MAAE,S;K;IAJtC,wC;MAEgB,Q;  
MADZ,IAAI,8CAAJ,C;QACI,OAA4C,CAApC,2EAAoC,kBAAQ,QAAR,C;OAEhD,OAAO,uBAAmB,SAAnB,E  
AAyB,gBAAZB,EAAiC,QAAjC,C;K;IAGX,4B;MAYiB,Q;MAFb,YAAY,gB;MACZ,YAAY,gB;MACC,2B;MAA  
b,OAAa,cAAb,C;QAAa,sB;QACT,KAAM,WAAI,IAAK,MAAT,C;QACN,KAAM,WAAI,IAAK,OAAT,C;;MAE  
V,OAAO,UAAS,KAAT,C;K;IAGX,+B;MAQqD,6BAAS,4BAAT,C;K;IAW0B,+G;MAAA,wC;MAAA,6B;MAAA  
,yB;MAAA,0C;MAAA,4C;MAAA,0B;MAAA,kC;K;;;mDAAA,Y;;;;kCAC9D,0C;cACb,gB;;;;cAAA,IAAO,iBPy  
FkD,UOzFzD,C;gBAAA,gB;;;cACI,QAAQ,yBAAO,iBAAQ,iBAAO,KAAf,C;cACf,WAAkB,WAAP,iBAAO,C;c  
ACIB,YAAgB,IAAI,iBAAO,KAAf,GAAqB,iBAAO,aAAI,CAAJ,EAAO,IAAP,CAA5B,GAA8C,I;cAC1D,gB;8BA  
AA,iCAAM,KAAN,O;kBAAA,2C;uBAAA,yB;cAAA,Q;;cAJJ,gB;;;cAMJ,W;;;;K;IAR+E,4D;MAAA,yD;u  
BAAA,mG;YAAA,S;iBAAA,Q;;iBAAA,uB;O;K;IAT/E,uC;MASmE,gBAAY,kCAAZ,C;K;IAkBhC,0D;MAE/B,w  
B;QAAA,WAAgC,I;MADhC,0B;MACA,0B;MACA,4B;K;IAGuC,0E;MAAA,oD;MACnC,gBA Ae,iCAAS,W;MA  
CxB,iBAAqB,E;MACrB,gBAAmB,I;K;oEAEnB,Y;MACI,OAAO,aAAS,UAAhB,C;QACI,WAAW,aAAS,O;QAC  
pB,IAAI,wCAAU,IAAV,MAAmB,sCAAvB,C;UACI,gBAAW,I;UACX,iBAAY,C;UACZ,M;;MAGR,iBAAY,C;K;  
8DAGhB,Y;MASW,Q;MARP,IAAI,mBAAa,EAAjB,C;QACI,iB;MACJ,IAAI,mBAAa,CAAjB,C;QACI,MAAM,6  
B;MACV,aAAa,a;MACb,gBAAW,I;MACX,iBAAY,E;MAEZ,OAAO,yE;K;iEAGX,Y;MACI,IAAI,mBAAa,EAAj  
B,C;QACI,iB;MACJ,OAAO,mBAAa,C;K;;2CAhC5B,Y;MAAuC,yD;K;;IA2C3C,qD;MAAY,0B;MAAmC,gC;K;I  
ACJ,gF;MAAA,0D;MACnC,gBA Ae,oCAAS,W;K;iEACxB,Y;MACI,OAAO,6CAA Y,aAAS,OAAR,C;K;oEAGX,  
Y;MACI,OAAO,aAAS,U;K;;8CAPxB,Y;MAAuC,4D;K;qDAWvC,oB;MACI,OAAO,uBAA4B,eAA5B,EAAcC,kB  
AAtC,EAAmD,QAA nD,C;K;;IAUf,4D;MAAY,0B;MAAmC,gC;K;IACJ,8F;MAAA,wE;MACnC,gBA Ae,2CAAS,  
W;MACxB,aAAY,C;K;wEACZ,Y;MAC0C,Q;MAAtC,OAAO,oDAAY,oBAAmB,iBAAnB,EAAmB,yBAAnB,QA  
AZ,EAAyC,aAAS,OAAD,C;K;2EAGX,Y;MACI,OAAO,aAAS,U;K;;qDARxB,Y;MAAuC,mE;K;;IAk3C,oC;M  
AAY,0B;K;IAC6C,wE;MACjD,gBA Ae,gCAAS,W;MACxB,aAAY,C;K;6DACZ,Y;MAC2C,Q;MAAvC,OAAO,iB  
AAa,oBAAmB,iBAAnB,EAAmB,yBAAnB,QAAb,EAA0C,aAAS,OAAnD,C;K;gEAGX,Y;MACI,OAAO,aAAS,U  
;K;;0CARxB,Y;MAAqD,wD;K;;IAmBzD,0D;MACI,4B;MACA,4B;MACA,4B;K;IAEuC,sE;MAAA,gD;MACnC,i  
BAAgB,gCAAU,W;MAC1B,iBAAgB,gCAAU,W;K;4DAC1B,Y;MACI,OAAO,sCAAU,cAAU,OAAPB,EAA4B,c  
AAU,OAAT,C;K;+DAGX,Y;MACI,OAAO,cAAU,UA AV,IAAuB,cAAU,U;K;;yCARhD,Y;MAAuC,uD;K;;IAc3  
C,6D;MACI,0B;MACA,gC;MACA,0B;K;IAEuC,4E;MAAA,sD;MACnC,gBA Ae,kCAAS,W;MACxB,oBAAiC,I;K  
;+DAEjC,Y;MACI,IAAI,CAAC,2BAAL,C;QACI,MAAM,6B;MACV,OAAO,gCA Ae,O;K;KEAG1B,Y;MACI,OA  
AO,2B;K;+EAGX,Y;MACQ,Q;MAAJ,IAAI,iEAA2B,KAA/B,C;QACI,oBAAe,I;MAEnB,OAAO,yBAAP,C;QACI  
,IAAI,CAAC,aAAS,UAA d,C;UACI,OAAO,K;;UAEP,cAAc,aAAS,O;UACvB,uBAAuB,wCAAS,2CAAY,OAAZ,  
CAAT,C;UACvB,IAAI,gBAAiB,UAArB,C;YACI,oBAAe,gB;YACf,OAAO,I;;MAInB,OAAO,I;K;;4CA9Bf,Y;M  
AAuC,0D;K;;IAoC9B,6I;MAAA,wC;MAAA,6B;MAAA,yB;MAAA,4C;MAAA,kD;MAAA,gD;MAAA,wB;MAA  
A,yB;MAAA,kC;K;;;yDAAA,Y;;;;kBAGyC,I;iCAFIC,C;cACI,sD;cAAhB,gB;;;;cAAA,KAAgB,yBAAhB,C;gBA  
AA,gB;;;cAAgB,oC;cACZ,aAAa,6BAAU,oBAAmB,uBAAnB,EAAmB,+BAAnB,QAAV,EAAuC,OAAvC,C;cAC  
b,gB;8BAAA,sCAAS,4BAAS,MAAT,CAAT,O;kBAAA,2C;uBAAA,yB;cAAA,Q;;cAFJ,gB;;;cAIJ,W;;;;K;I  
ANS,0F;MAAA,yD;uBAAA,iI;YAAA,S;iBAAA,Q;;iBAAA,uB;O;K;IADb,wD;MACI,gBAAS,kDAAT,C;K;;IAo  
ByB,qD;MACzB,0B;MACA,8B;MACA,0B;MC3TA,IAAI,ED+TQ,qBAAc,CC/TtB,CAAJ,C;QACI,cD8T2B,+CA  
A4C,iB;QC7TvE,MAAM,gCAAyB,OAAQ,WAAjC,C;OAFV,IAAI,EDgUQ,mBAAyB,CChUpB,CAAJ,C;QACI,gB  
D+TyB,6CAA0C,e;QC9TnE,MAAM,gCAAyB,SAAQ,WAAjC,C;OAFV,IAAI,EDiUQ,mBAAyB,iBCjUpB,CAAJ,  
C;QACI,gBDgUkC,0DAAuD,eAAvD,WAAmE,iB;QC/TrG,MAAM,gCAAyB,SAAQ,WAAjC,C;Q;SFDkUa,Y;MA  
AQ,yBAAW,iBAAX,I;K;yCAE/B,a;MAAyC,OAAL,KAAK,YAAT,GAAGB,eAAhB,GAAqC,gBAAyB,eAAZ,EAA  
S B,oBAAa,CAAb,IAAtB,EAAcC,eAAAT,C;K;yCAC9E,a;MAAyC,OAAL,KAAK,YAAT,GAAGB,IAAhB,GAA0B,g  
BAAyB,eAAZ,EAAcB,iBAAtB,EAAkC,oBAAa,CAAb,IAAIC,C;K;IAEzC,8D;MAAA,wC;MAEtB,gBA Ae,2BAAS  
,W;MACxB,gBA Ae,C;K;0DAEf,Y;MAEI,OAAO,gBAAW,kCAAX,IAAyB,aAAS,UAAzC,C;QACI,aAAS,O;QAC  
T,qC;;K;2DAIR,Y;MACI,a;MACA,OAAQ,gBAAW,gCAAZ,IAAyB,aAAS,U;K;wDAG7C,Y;MACI,a;MACA,IA  
AI,iBAAY,gCAAhB,C;QACI,MAAM,6B;MACV,qC;MACA,OAAO,aAAS,O;K;;qCAvBxB,Y;MAA0B,mD;K;;IA  
gCA,uC;MAC1B,0B;MACA,oB;MC3WA,IAAI,ED+WQ,gBAAS,CC/WjB,CAAJ,C;QACI,cD8WsB,yCAAsC,YA  
AtC,M;QC7WtB,MAAM,gCAAyB,OAAQ,WAAjC,C;Q;0CDgXV,a;MAAyC,OAAL,KAAK,YAAT,GAAGB,eAAh

B,GAAqC,gBAAY,eAAZ,EAAsB,CAAtB,EAAYB,YAAzB,C;K;0CAC9E,a;MAAyC,OAAI,KAAK,YAAT,GAAg  
B,IAAhB,GAA0B,iBAAa,eAAb,EAAB,CAAvB,C;K;IAE5B,gE;MACnC,YAAW,yB;MACX,gBA Ae,4BAAS,W;  
K;yDAExB,Y;MACI,IAAI,cAAQ,CAAZ,C;QACI,MAAM,6B;MACV,6B;MACA,OAAO,aAAS,O;K;4DAGpB,Y;  
MACI,OAAO,YAAO,CAAP,IAAY,aAAS,U;K;;sCAZpC,Y;MAAuC,oD;K;;IAsB3C,gD;MACI,0B;MACA,4B;K;I  
AEuC,0E;MAAA,oD;MACnC,gBA Ae,iCAAS,W;MACxB,iBAAqB,E;MACrB,gBAAmB,I;K;oEAEnB,Y;MACI,I  
AAI,aAAS,UAAb,C;QACI,WAAW,aAAS,O;QACpB,IAAI,wCAAU,IAAV,CAAJ,C;UACI,iBAAY,C;UACZ,gBA  
AW,I;UACX,M;UAGR,iBAAY,C;K;8DAGhB,Y;MAMiB,Q;MALb,IAAI,mBAAa,EAAjB,C;QACI,iB;MACJ,IAA  
I,mBAAa,CAAJB,C;QACI,MAAM,6B;MACV,aACa,gF;MAGb,gBAAW,I;MACX,iBAAY,E;MACZ,OAAO,M;K;  
iEAGX,Y;MACI,IAAI,mBAAa,EAAjB,C;QACI,iB;MACJ,OAAO,mBAAa,C;K;;2CAIC5B,Y;MAAuC,yD;K;;IA2  
Cb,uC;MAC1B,0B;MACA,oB;MC5bA,IAAI,ED+bQ,gBAAS,CC/bjB,CAAJ,C;QACI,cD8bsB,yCAAsC,YAAtC,M  
;QC7btB,MAAM,gCAAYB,OAAQ,WAAjC,C;Q;0CDgcV,a;MItXO,SJsXmC,eAAQ,CAAR,I;MAAD,OAA4B,KA  
AK,CAAT,GAAY,yBAAZ,GAAuC,iBAAa,eAAb,EAAB,CAAvB,C;K;0CACxG,a;MIvXO,SJuXmC,eAAQ,CAA  
R,I;MAAD,OAA4B,KAAK,CAAT,GAAY,yBAAZ,GAAuC,gBAAY,eAAZ,EAAsB,YAAtB,EAA6B,EAA7B,C;K;  
IAEjE,gE;MACnC,gBA Ae,4BAAS,W;MACxB,YAAW,yB;K;2DAEX,Y;MAEI,OAAO,YAAO,CAAP,IAAY,aAA  
S,UAA5B,C;QACI,aAAS,O;QACT,6B;;K;yDAIR,Y;MACI,a;MACA,OAAO,aAAS,O;K;4DAGpB,Y;MACI,a;MA  
CA,OAAO,aAAS,U;K;;sCAnBxB,Y;MAAuC,oD;K;;IA6B3C,gD;MACI,0B;MACA,4B;K;IAGuC,0E;MAAA,oD;  
MACnC,gBA Ae,iCAAS,W;MACxB,iBAAqB,E;MACrB,gBAAmB,I;K;gEAEnB,Y;MACI,OAAO,aAAS,UAAhB,  
C;QACI,WAAW,aAAS,O;QACpB,IAAI,CAAC,wCAAU,IAAV,CAAL,C;UACI,gBAAW,I;UACX,iBAAY,C;UA  
CZ,M;;MAGR,iBAAY,C;K;8DAGhB,Y;MAMqB,Q;MALjB,IAAI,mBAAa,EAAjB,C;QACI,a;MAEJ,IAAI,mBAA  
a,CAAJB,C;QACI,aACa,gF;QACb,gBAAW,I;QACX,iBAAY,C;QACZ,OAAO,M;OAEX,OAAO,aAAS,O;K;iEAG  
pB,Y;MACI,IAAI,mBAAa,EAAjB,C;QACI,a;MACJ,OAAO,mBAAa,CAAb,IAAkB,aAAS,U;K;;2CAIC1C,Y;MA  
AuC,yD;K;;IAuCN,+C;MAAC,sB;MAAiC,gC;K;0CACnE,Y;MAAuC,4BAAiB,aAAO,WAAxB,EAAoC,kBAAPC  
,C;K;;IAGP,+C;MAAuE,2B;MAAtE,sB;MAAiC,gC;MACIE,kBAAuB,c;K;6CAEvB,Y;MACI,OAAO,aAAO,UAA  
d,C;QACI,WAAW,aAAO,O;QACIB,UAAU,mBAAY,IAAZ,C;QAEV,IAAI,eAAS,WAAI,GA AJ,CAAb,C;UACI,  
mBAAQ,IAAR,C;UACA,M;;MAIR,W;K;;IAKgC,0D;MAAC,wC;MAAuC,kC;K;IACrC,0E;MAAA,oD;MACnC,g  
BAAmB,I;MACnB,iBAAqB,E;K;oEAErB,Y;MACI,gBA Ae,mBAAa,EAAjB,GAAqB,+CAArB,GAA4C,2CAAa,4  
BAAb,C;MACvD,iBAAGb,qBAAJ,GAAsB,CAAtB,GAA6B,C;K;8DAG7C,Y;MAMiB,Q;MALb,IAAI,iBAAY,CA  
AhB,C;QACI,iB;MAEJ,IAAI,mBAAa,CAAJB,C;QACI,MAAM,6B;MACV,aAAa,8D;MAEb,iBAAY,E;MACZ,O  
AAO,M;K;iEAGX,Y;MACI,IAAI,iBAAY,CAAhB,C;QACI,iB;MACJ,OAAO,mBAAa,C;K;;2CAx5B,Y;MAAuC  
,yD;K;;IA6B3C,kC;MAWI,OAAW,iDAAJ,GAAwC,SAAxC,GAakD,4BAAwB,SAAxB,C;K;IAelB,uD;MAAA,q  
B;QAAE,6B;O;K;IAX7C,wC;MAWI,OAA2D,cAApD,sBAAkB,YAAIB,EAAGC,qCAAhC,CAAoD,C;K;IAqBrC,i  
D;MAAA,mB;QAAE,mB;O;K;IAIB5B,gD;MAeI,OAAI,YAAJ,GACI,2BADJ,GAGI,sBAAkB,+BAAIB,EAA4B,Y  
AA5B,C;K;IAER,wD;MAcI,6BAAkB,YAAIB,EAAGC,YAAhC,C;K;ILxpBJ,oB;MAAA,wB;MACI,8C;K;gCAEA,  
iB;MAA4C,oCAAmB,KAAM,U;K;kCACrE,Y;MAA+B,Q;K;kCAC/B,Y;MAAkC,W;K;gFAEX,Y;MAAQ,Q;K;iC  
AC/B,Y;MAAkC,W;K;wCACIC,mB;MAAmD,Y;K;6CACnD,oB;MAAmE,OAAA,QAAS,U;K;kCAE5E,Y;MAA6  
C,kC;K;uCAE7C,Y;MAAiC,6B;K;;IADrC,gC;MAAA,+B;QAAA,c;OAAA,wB;K;IAkBA,oB;MAIoC,6B;K;IAEp  
C,2B;MAMmD,OAAI,QAAS,OAAT,GAAgB,CAApB,GAAgC,MAAT,QAAS,CAAhC,GAA6C,U;K;iFAEHg,yB;  
MAAA,mD;MAAA,mB;QAKwC,iB;O;KALxC,C;6FAOA,yB;MAAA,uE;MAAA,mB;QAQsD,2B;O;KARtD,C;IA  
UA,kC;MAKiE,OAAS,aAAT,QAAS,EAAa,qBAAc,YAAY,QAAS,OAArB,CAAd,CAAb,C;K;uFAE1E,yB;MAA  
A,2D;MAAA,mB;QAGgD,qB;O;KAHhD,C;IAKA,+B;MAC2D,OAAS,aAAT,QAAS,EAAa,eAAQ,YAAY,QAAS,  
OAArB,CAAR,CAAb,C;K;2FAEpE,yB;MAAA,uE;MAAA,mB;QAMwD,2B;O;KANxD,C;IAQA,iC;MAKME,OA  
AS,aAAT,QAAS,EAAa,qBAAc,YAAY,QAAS,OAArB,CAAd,CAAb,C;K;IAE5E,+B;MAMyD,OAAI,eAAJ,GAA  
qB,MAAM,OAAO,CAArB,GAAyC,U;K;IAEIG,kC;MAQI,OAAgB,gBAAT,QAAS,EAAgB,sBAAhB,C;K;sFAGp  
B,yB;MavBA,uE;MbuBA,gC;QanB8B,gBAAnB,oB;QbqCiB,aS/CxB,W;QT+CA,OS9CO,SISwC,Q;O;KbmBnD,C;  
wFA0BA,yB;Ma1CA,wE;Mb0CA,0C;QatCsC,gBAA3B,mBb4DiB,Qa5DjB,C;Qb4D2B,aS7EiC,W;QT6EA,OS5E  
O,SIgBgD,Q;O;KbsC3D,C;sFA+BA,yB;MAAA,mD;MAAA,4B;QAEkD,uCAAQ,U;O;KAF1D,C;IAIA,wC;MAAg  
D,QAAM,cAAN,C;aAC5C,C;UAD4C,OACvC,U;aACL,C;UAF4C,OAEvC,MAAM,oBAAW,OAAjB,C;gBAFuC,  
OAGpC,S;;K;IKnKZ,oD;MAQuF,wC;K;IARvF,8CASI,Y;MAAuC,8B;K;IAT3C,gF;IyKLA,yC;MxK4BI,IAAI,Ew

K3BI,OAAO,CAAP,IAAY,OAAO,CxK2BvB,CAAJ,C;QACI,cwK3BI,aAAJ,GACI,yEADJ,GAGI,8C;QxKyBJ,M  
AAM,gCAAYB,OAAQ,WAAjC,C;Q;IwKnBM,mI;MAAA,mB;QAAE,wBAAiB,gCAAjB,EAA6B,YAA7B,EAAM  
C,YAAnC,EAAYC,sBAAzC,EAAYD,mBAAzD,C;O;K;IAFtB,gF;MACI,oBAAoB,IAApB,EAA0B,IAA1B,C;MAC  
A,oCAAgB,6EAAhB,C;K;IAKyB,yL;MAAA,wC;MAAA,6B;MAAA,yB;MAAA,wC;MAAA,wC;MAAA,gD;MA  
AA,sD;MAAA,4D;MAAA,wB;MAAA,0B;MAAA,uB;MAAA,0B;MAAA,wB;MAAA,qB;MAAA,4B;MAAA,kC;  
K;;;2DAAA,Y;;;;cACrB,4BAAiC,eAAL,uBAAK,EAAa,IAAb,C;+BACvB,0BAAO,uBAAP,I;cACV,IAAI,kBAA  
O,CAAX,C;oCACiB,iBAAa,qBAAb,C;kCACF,C;gBACD,6C;gBAAV,iB;;;sCAaa,gBAAc,qBAAd,C;gBACH,+C;  
gBAAV,gB;;;;;;cAAA,KAAU,2BAAV,C;gBAAA,gB;;;cAAU,kC;cACN,mBAAO,WAAI,GAAJ,C;cACP,IAAI,m  
BAAO,SAAX,C;gBACI,IAAI,mBAAO,KAAP,GAAc,uBAAiB,C;kBAA0B,sBAAS,mBAAO,kBAAuB,uBAAvB,  
C;kBAA8B,gB;;;kBAAxE,gB;;;gBADJ,gB;;;cAGI,gB;8BAAA,iCAAU,8BAAJ,GAAiB,mBAAjB,GAA6B,iBAA  
U,mBAAV,CAAnC,O;kBAAA,2C;uBAAA,yB;cAAA,Q;cACA,mBAAO,qBAAy,uBAAZ,C;cAJX,gB;;;cAFJ,gB;  
;cASA,IAAI,iCAAJ,C;gBACI,gB;;;gBADJ,iB;;;cACI,IAAO,mBAAO,KAAd,IAAqB,uBAArB,C;gBAAA,gB;;;c  
ACI,gB;8BAAA,iCAAU,8BAAJ,GAAiB,mBAAjB,GAA6B,iBAAU,mBAAV,CAAnC,O;kBAAA,2C;uBAAA,yB;  
cAAA,Q;cACA,mBAAO,qBAAy,uBAAZ,C;cAFX,gB;;;cAIA,IhL4K4C,CgL5KxC,mBhL4KyC,UgL5K7C,C;gB  
AAyB,iB;gCAAA,iCAAM,mBAAN,O;oBAAA,2C;yBAAA,yB;gBAAA,Q;gBAAzB,iB;;;cAjCR,W;cA4BI,iB;;;  
cA1BJ,iB;;;cAGI,KAAU,yBAAV,C;gBAAA,iB;;;6BAAU,sB;cACN,IAAI,kBAAO,CAAX,C;gBAAgB,oCAAQ,C  
AAR,I;gBAAW,iB;;;gBAA3B,iB;;;cACA,iBAAO,WAAI,YAAJ,C;cACP,IAAI,iBAAO,KAAP,KAAe,uBAAnB,C  
;gBACI,iB;gCAAA,iCAAM,iBAAN,O;oBAAA,2C;yBAAA,yB;gBAAA,Q;gBADJ,iB;;;cAEI,IAAI,8BAAJ,C;gB  
AAiB,iBAAO,Q;gBAAa,oBAAS,iBAAU,uBAAV,C;cAC9C,kBAAO,c;cAHX,iB;;;cAHJ,iB;;;cASA,IhL+LgD,Cg  
L/L5C,iBhL+L6C,UgL/LjD,C;gBACI,IAAI,qCAAkB,iBAAO,KAAP,KAAe,uBAArC,C;kBAA2C,iB;kCAAA,iCA  
AM,iBAAN,O;sBAAA,2C;2BAAA,yB;kBAAA,Q;kBAA3C,iB;;;gBADJ,iB;;;cAdJ,W;cAcI,iB;cAZJ,iB;cAk  
CJ,W;;;;;;K;IARCyB,sI;MAAA,yD;uBAAA,6K;YAAA,S;iBAAA,Q;iBAAA,uB;O;K;IAF7B,6E;MACI,IAAI,  
CAAC,QAAS,UAAc,C;QAAyB,OAAO,2B;MACHC,OAAO,WAAkB,0EAAiB,C;K;IAwCwB,6B;MAA8B,uB;MA  
A7B,kB;MACHC,mBAA6B,C;MAC7B,eAAyB,C;K;2CAEzB,8B;MACI,+DAAkB,SAAlB,EAA6B,OAA7B,EAAs  
C,WAAK,KAA3C,C;MACA,mBAAiB,S;MACjB,eAAa,UAAU,SAAV,I;K;0CAGjB,iB;MACI,+DAAkB,KAAiB,E  
AAyB,YAAzB,C;MAEA,OAAO,wBAAK,mBAAy,KAAZ,IAAL,C;K;qFAGY,Y;MAAQ,mB;K;;IASR,wC;MAAq  
D,uB;MAApD,sB;MxKrDxB,IAAI,EwKuDQ,cAAc,CxKvDtB,CAAJ,C;QACI,cwKsD2B,wE;QxKrD3B,MAAM,g  
CAAYB,OAAQ,WAAjC,C;OAFV,IAAI,EwKwDQ,cAAc,aAAO,OxKxD7B,CAAJ,C;QACI,gBwKuDqC,wFAA+E  
,aAAO,O;QxKtD3H,MAAM,gCAAYB,SAAQ,WAAjC,C;OwK2DV,kBAAuB,aAAO,O;MAC9B,oBAA8B,C;MAE  
9B,sBAAyB,U;K;kFAAzB,Y;MAAA,0B;K,OAAA,gB;MAAA,0B;K;uCAGA,iB;MAGW,Q;MAFP,+DAAkB,KA  
AlB,EAAyB,SAAzB,C;MAEA,OAAO,sBAAGmC,CAnG5B,iBAmG6B,GAnGV,KAmGU,IAAD,IAAa,eAnGhD,4  
D;K;kCAGX,Y;MAAe,qBAAQ,e;K;IAEgB,4D;MAAA,sC;MAAS,2B;MAC5C,eAAoB,oB;MACpB,eAAoB,4B;K;  
8DAEpB,Y;MAKgB,Q;MAJZ,IAAI,iBAAS,CAAb,C;QACI,W;;QAGA,mBAAQ,sCAAQ,YAAP,4DAAR,C;QAC  
A,eAoFkC,CAPf1B,YAoF2B,GAPfB,CAoFa,IAAD,IAAa,+B;QAnF/C,mC;;K;;oCAXZ,Y;MAAuC,kD;K;2CAGv  
C,iB;MAGiE,UAQ1C,MAR0C,EAe1C,MAf0C,EAqBtD,M;MAiBP,aACQ,KAAM,OAAN,GAAa,IAAK,KAAtB,G  
AAkC,UAAN,KAAM,EAAO,IAAK,KAAZ,CAAIC,GAAyD,kD;MAE7D,WAAW,IAAK,K;MAEHb,WAAW,C;M  
ACX,UAAU,iB;MAEV,OAAO,OAAO,IAAP,IAAe,MAAM,eAA5B,C;QACI,OAAO,IAAP,IAAe,wBAAO,GAAP,  
gE;QACf,mB;QACA,iB;;MAGJ,MAAM,C;MACN,OAAO,OAAO,IAAd,C;QACI,OAAO,IAAP,IAAe,wBAAO,G  
AAP,gE;QACf,mB;QACA,iB;;MAEJ,IAAI,MAAO,OAAP,GAAc,IAAK,KAAvB,C;QAA6B,OAAO,IAAK,KAAZ,  
IAAoB,I;MAEjD,OAAO,uD;K;mCAGX,Y;MACI,OAAO,qBAAQ,gBAAa,SAAb,OAAR,C;K;4CAGX,uB;MAKI,  
kBAAoD,eAAjC,mBAAy,mBAAa,CAAzB,IAA8B,CAA9B,IAAiC,EAAa,WAAb,C;MACpD,gBAAoB,sBAAc,C  
AAiB,GAA4B,UAAP,aAAO,EAAO,WAAP,CAA5B,GAAqD,qBAAQ,gBAAa,WAAb,OAAR,C;MACrE,OAAO,e  
AAW,SAAX,EAAS,SAAtB,C;K;qCAGX,mB;MAII,IAAI,aAAJ,C;QACI,MAAM,6BAASB,qBAAtB,C;OAGV,c  
A6B0C,CA7BnC,iBA6BoC,GA7BjB,SA6BiB,IAAD,IAAa,eA7BvD,IAAmC,O;MACnC,6B;K;+CAGJ,a;MxKhJA,  
IAAI,EwKoJQ,KAAK,CxKpJb,CAAJ,C;QACI,cwKmJkB,wC;QxKlJIB,MAAM,gCAAYB,OAAQ,WAAjC,C;OAF  
V,IAAI,EwKqJQ,KAAK,SxKrJb,CAAJ,C;QACI,gBwKoJqB,wEAA8D,S;QxKnJnF,MAAM,gCAAYB,SAAQ,WA  
AjC,C;OwKqJn,IAAI,IAAI,CAAR,C;QACI,YAAY,iB;QACZ,UAGBsC,CAhB5B,KAGb6B,GAhBf,CAGBe,IAAD,  
IAAa,e;QAdnD,IAAI,QAAQ,GAAZ,C;UACW,OAAP,aAAO,EAAK,IAAL,EAAW,KAAX,EAAkB,eAAiB,C;UA

CA,OAAP,aAAO,EAAK,IAAL,EAAW,CAAX,EAAc,GAAd,C;;UAEA,OAAP,aAAO,EAAK,IAAL,EAAW,KAA X,EAAkB,GAAlB,C;;QAGX,oBAAa,G;QACb,wBAAQ,CAAR,I;Q;qCAKR,wB;MAC8C,QAAC,YAAO,CAAP,I AAD,IAAa,e;K;;IA9G3D,0C;MAAA,oD;MAA6B,uBAAK,gBAAmB,QAAnB,OAAL,EAAmC,CAAnC,C;MAA7 B,Y;K;ICvFJ,0C;MAII,QAAQ,I;MACR,QAAQ,K;MACR,YAAY,kBAAM,CAAC,OAAO,KAAP,IAAD,IAAiB,C AAjB,IAAN,C;MACZ,OAAO,KAAK,CAAZ,C;QACI,OrL+B4E,0BqL/BrE,kBAAM,CAAN,CrL0Q2B,KAAL,GA AiB,GA3O8B,EqL/B1D,KrL0QgB,KAAL,GAAiB,GA3O8B,CqL/BrE,IAAP,C;UACI,a;;QACJ,OrL6B4E,0BqL7Br E,kBAAM,CAAN,CrLwQ2B,KAAL,GAAiB,GA3O8B,EqL7B1D,KrLwQgB,KAAL,GAAiB,GA3O8B,CqL7BrE,I AAP,C;UACI,a;;QACJ,IAAI,KAAK,CAAT,C;UACI,UAAU,kBAAM,CAAN,C;UACV,kBAAM,CAAN,EAAW,k BAAM,CAAN,CAAX,C;UACA,kBAAM,CAAN,EAAW,GAAX,C;UACA,a;UACA,a;;MAGR,OAAO,C;K;IAGX, uC;MAGI,YAAY,aAAU,KAAV,EAAiB,IAAjB,EAAuB,KAAvB,C;MACZ,IAAI,QAAO,QAAQ,CAAR,IAAP,CA AJ,C;QACI,UAAU,KAAV,EAAiB,IAAjB,EAAuB,QAAQ,CAAR,IAAvB,C;MACJ,IAAI,QAAQ,KAAZ,C;QACI, UAAU,KAAV,EAAiB,KAAjB,EAAwB,KAAxB,C;K;IAGR,0C;MAII,QAAQ,I;MACR,QAAQ,K;MACR,YAAY,k BAAM,CAAC,OAAO,KAAP,IAAD,IAAiB,CAAjB,IAAN,C;MACZ,OAAO,KAAK,CAAZ,C;QACI,OnLM6E,0B mLNtE,kBAAM,CAAN,CnL0O2B,KAAL,GAAiB,KApO+B,EmLN3D,KnL0OgB,KAAL,GAAiB,KApO+B,CmL NtE,IAAP,C;UACI,a;;QACJ,OnLI6E,0BmLJtE,kBAAM,CAAN,CnLwO2B,KAAL,GAAiB,KApO+B,EmLJ3D,Kn LwOgB,KAAL,GAAiB,KApO+B,CmLJtE,IAAP,C;UACI,a;;QACJ,IAAI,KAAK,CAAT,C;UACI,UAAU,kBAAM, CAAN,C;UACV,kBAAM,CAAN,EAAW,kBAAM,CAAN,CAAX,C;UACA,kBAAM,CAAN,EAAW,GAAX,C;UA CA,a;UACA,a;;MAGR,OAAO,C;K;IAGX,yC;MAGI,YAAY,aAAU,KAAV,EAAiB,IAAjB,EAAuB,KAAvB,C;M ACZ,IAAI,QAAO,QAAQ,CAAR,IAAP,CAAJ,C;QACI,YAAU,KAAV,EAAiB,IAAjB,EAAuB,QAAQ,CAAR,IAA vB,C;MACJ,IAAI,QAAQ,KAAZ,C;QACI,YAAU,KAAV,EAAiB,KAAjB,EAAwB,KAAxB,C;K;IAGR,0C;MAII, QAAQ,I;MACR,QAAQ,K;MACR,YAAY,kBAAM,CAAC,OAAO,KAAP,IAAD,IAAiB,CAAjB,IAAN,C;MACZ, OAAO,KAAK,CAAZ,C;QACI,OpLnB8D,YoLmBvD,kBAAM,CAAN,CpLnBwE,KAAjB,EoLmB5C,KpLnByE,K AA7B,CoLmBvD,IAAP,C;UACI,a;;QACJ,OpLrB8D,YoLqBvD,kBAAM,CAAN,CpLrBwE,KAAjB,EoLqB5C,Kp LrByE,KAA7B,CoLqBvD,IAAP,C;UACI,a;;QACJ,IAAI,KAAK,CAAT,C;UACI,UAAU,kBAAM,CAAN,C;UAC V,kBAAM,CAAN,EAAW,kBAAM,CAAN,CAAX,C;UACA,kBAAM,CAAN,EAAW,GAAX,C;UACA,a;UACA,a;;MAGR,OAAO,C;K;IAGX,yC;MAGI,YAAY,aAAU,KAAV,EAAiB,IAAjB,EAAuB,KAAvB,C;MACZ,IAAI,QAA O,QAAQ,CAAR,IAAP,CAAJ,C;QACI,YAAU,KAAV,EAAiB,IAAjB,EAAuB,QAAQ,CAAR,IAAvB,C;MACJ,IA AI,QAAQ,KAAZ,C;QACI,YAAU,KAAV,EAAiB,KAAjB,EAAwB,KAAxB,C;K;IAGR,0C;MAII,QAAQ,I;MACR ,QAAQ,K;MACR,YAAY,kBAAM,CAAC,OAAO,KAAP,IAAD,IAAiB,CAAjB,IAAN,C;MACZ,OAAO,KAAK,C AAZ,C;QACI,OpK5C+D,aoK4CxD,kBAAM,CAAN,CpK5C0E,KAAIB,EoK4C7C,KpK5C2E,KAA9B,CoK4CxD, IAAP,C;UACI,a;;QACJ,OpK9C+D,aoK8CxD,kBAAM,CAAN,CpK9C0E,KAAIB,EoK8C7C,KpK9C2E,KAA9B,C oK8CxD,IAAP,C;UACI,a;;QACJ,IAAI,KAAK,CAAT,C;UACI,UAAU,kBAAM,CAAN,C;UACV,kBAAM,CAAN, EAAW,kBAAM,CAAN,CAAX,C;UACA,kBAAM,CAAN,EAAW,GAAX,C;UACA,a;UACA,a;;MAGR,OAAO,C; K;IAGX,yC;MAGI,YAAY,aAAU,KAAV,EAAiB,IAAjB,EAAuB,KAAvB,C;MACZ,IAAI,QAAO,QAAQ,CAAR,I AAP,CAAJ,C;QACI,YAAU,KAAV,EAAiB,IAAjB,EAAuB,QAAQ,CAAR,IAAvB,C;MACJ,IAAI,QAAQ,KAAZ, C;QACI,YAAU,KAAV,EAAiB,KAAjB,EAAwB,KAAxB,C;K;IAKR,gD;MAI6E,UAAU,KAAV,EAAiB,SAAjB,E AA4B,UAAU,CAAV,IAA5B,C;K;IAC7E,gD;MAC6E,YAAU,KAAV,EAAiB,SAAjB,EAA4B,UAAU,CAAV,IAA 5B,C;K;IAC7E,gD;MAC6E,YAAU,KAAV,EAAiB,SAAjB,EAA4B,UAAU,CAAV,IAA5B,C;K;IAC7E,gD;MAC6 E,YAAU,KAAV,EAAiB,SAAjB,EAA4B,UAAU,CAAV,IAA5B,C;K;IvK9I7E,0C;MF0BI,IAAI,EEjBI,SAAU,OA AV,GAAiB,CfiBrB,CAAJ,C;QACI,cAda,qB;QAeb,MAAM,gCAAYB,OAAQ,WAAjC,C;OEIBV,OAAO,oBAAoB ,CAApB,EAAuB,CAAvB,EAA0B,SAA1B,C;K;IAGX,8C;MACe,Q;MAAX,wBAAW,SAAX,gB;QAAW,SAAA,S AAX,M;QACI,SAAS,GAAG,CAAH,C;QACT,SAAS,GAAG,CAAH,C;QACT,WAAW,cAAc,EAAc,EAAkB,EAA IB,C;QACX,IAAI,SAAQ,CAAZ,C;UAAe,OAAO,I;;MAE1B,OAAO,C;K;sGAGX,yB;MAAA,8D;MAAA,iC;QASI ,OAAO,cAAc,SAAS,CAAT,CAAd,EAA2B,SAAS,CAAT,CAA3B,C;O;KATX,C;sGAYA,sC;MASI,OAAO,UAA W,SAAQ,SAAS,CAAT,CAAR,EAAqB,SAAS,CAAT,CAArB,C;K;IAAtB,6B;MAWY,Q;MALR,IAAI,MAAM,CA AV,C;QAAa,OAAO,C;MACpB,IAAI,SAAJ,C;QAAe,OAAO,E;MACTb,IAAI,SAAJ,C;QAAe,OAAO,C;MAGtB,O AA8B,iBAAtB,mDAAsB,EAAU,CAAV,C;K;IAAz,6C;MAAA,uB;QAAU,2BAAoB,CAApB,EAAuB,CAAvB,EA A0B,iBAA1B,C;O;K;IAVhC,8B;MF7CI,IAAI,EEsDI,SAAU,OAAV,GAAiB,CfTDrB,CAAJ,C;QACI,cAda,qB;QA



C;O;KANJ,C;2GAQA,yB;MAAA,4B;MDgDQ,kD;MChDR,uC;QAOI,6BDgDQ,WAAO,cChDW,SDgDX,CAAP,C  
ChDR,C;O;KAPJ,C;+FAUA,yB;MAAA,kC;MAAA,mD;MAAA,yE;QASI,sC;QAAA,4C;O;MATJ,iGAWY,Y;QA  
AQ,2B;OAXpB,E;MAAA,0DAaQ,kB;QACI,wBAAW,MAAX,C;O;MADz,sF;MAAA,sC;QASI,0D;O;KATJ,C;IAi  
BA,gD;MAaI,4BAA0D,YAAzC,wCAA6B,UAA7B,CAAyC,CAA1D,EAAyE,yBAAzE,C;K;IAEJ,4D;MAcI,4BAA  
oE,YAAAnD,0CAA6B,QAA7B,EAAuC,UAAvC,CAAmD,CAApE,EAAmF,yBAAAnF,C;K;IAEJ,+C;MAU6C,YAAz  
C,wCAA6B,UAA7B,CAAyC,CAtEzC,oBDgDQ,WcBsD,kBDtBtD,CChDR,C;K;IAyEJ,2D;MAWuD,YAAAnD,0C  
AA6B,QAA7B,EAAuC,UAAvC,CAAmD,CAPFnD,oBDgDQ,WCoCgE,kBDpChE,CChDR,C;K;IAuFJ,+C;MAYI,  
OAA6C,8BAAtC,c;K;8EAZX,yB;MAAA,oE;MAAA,6E;MAYiD,gD;QAAA,oB;UACzC,WAAW,sBAAmB,YAA  
F,CAAE,CAAnB,C;UACX,cAAM,IAAN,C;UADA,OAEA,IAAK,a;S;O;MAfb,sC;QAYW,mBAAsC,8BAAtC,6B;  
QAAP,OAAO,kD;O;KAZX,C;qGA0BI,yB;MAAA,2D;MAAA,mB;QACI,MAAM,6BAAoB,0BAApB,C;O;KADV  
,C;;M6HzIA,yC;;IAAA,uC;MAAA,2C;K;;IAAA,mD;MAAA,kD;QAAA,iC;OAAA,2C;K;+EAKBA,wB;K;oDAaA  
,e;MAK2C,IAAI,IAAJ,EAGK,M;MAL5C,IAAI,+CAAJ,C;QAEI,OAAW,GAAl,kBAAS,IAAK,IAAd,CAAR,GAA  
4B,cAAI,OAAJ,GAAl,iBAAQ,IAAR,CAAJ,yCAA5B,GAAyD,I;OAGpE,OAAW,8CAA4B,GAAhC,GAAqC,8EA  
ArC,GAAoD,I;K;yDAI/D,e;MAGI,IAAI,+CAAJ,C;QACI,OAAW,GAAl,kBAAS,IAAK,IAAd,CAAJ,IAA0B,GAA  
LiBAAQ,IAAR,CAAJ,QAA9B,GAAyD,mCAAzD,GAAoF,I;OAE/F,OAAW,8CAA4B,GAAhC,GAAqC,mCAArC  
,GAAgE,I;K;;;ICtChD,oD;MACf,cAAc,GAAl,kBAAS,OAAQ,IAAjB,C;MACIB,IAAI,YAAY,mCAAhB,C;QAD  
A,OACuC,O;;QAEnc,kBAakB,oBAAQ,yCAAR,C;QACIB,IAAI,mBAAJ,C;UAJJ,OAI6B,oBAAgB,OAAhB,EA  
AyB,OAAzB,C;;UACrB,WAAW,OAAQ,kBAAS,yCAAT,C;UAL3B,OAMY,SAAS,mCAAb,GAAoC,oBAAgB,O  
AAhB,EAAyB,WAAzB,CAApC,GACI,oBAAgB,oBAAgB,IAAhB,EAA5B,OAAtB,CAAhB,EAAGD,WAAhD,C;;;  
K;8CAdxB,mB;MAKI,OAAI,YAAY,mCAAhB,GAAuC,IAAvC,GACI,OAAQ,cAAK,IAAL,EAAW,4BAAx,C;K;  
;;;qDAiCZ,e;MAEyB,Q;MADrB,OACI,OAAA,IAAK,IAAL,EAAy,GAAZ,CAAJ,GAAqB,0EAArB,GAAoC,I;K;  
sDAExC,8B;MACI,iBAAU,OAAV,EAAmB,IAAnB,C;K;0DAEJ,e;MACI,OAAI,OAAA,IAAK,IAAL,EAAy,GAA  
Z,CAAJ,GAAqB,mCAArB,GAAgD,I;K;;IC1DP,8C;MAAC,wB;K;kFAAA,Y;MAAA,yB;K;;IAiCe,wD;MAEjE,k  
C;MAEA,4BAAqC,mDAAJ,GAAkD,OAAQ,qBAA1D,GAA0E,O;K;4DAE3G,mB;MAA6C,+BAAS,OAAT,C;K;6  
DAC7C,e;MAA8C,eAAQ,IAAR,IAAgB,8BAAe,G;K;;IAGjF,+C;MAW2C,IAAI,IAAJ,EAGV,M;MAL7B,IAAI,+  
CAAJ,C;QAEI,OAAW,GAAl,kBAAS,SAAK,IAAd,CAAR,GAA4B,cAAI,OAAJ,GAAl,iBAAQ,SAAR,CAAJ,yC  
AA5B,GAAyD,I;OAGpE,OAAW,SAAK,IAAL,KAAa,GAAjB,GAAsB,mFAAtB,GAAqC,I;K;IAGhD,6C;MAUI,I  
AAI,+CAAJ,C;QACI,OAAW,GAAl,kBAAS,SAAK,IAAd,CAAJ,IAA0B,GAAl,iBAAQ,SAAR,CAAJ,QAA9B,GA  
AyD,mCAAzD,GAAoF,S;OAE/F,OAAW,SAAK,IAAL,KAAa,GAAjB,GAAsB,mCAAtB,GAAiD,S;K;IAG5D,iC;  
MAAA,qC;MAKI,4B;K;oDACA,Y;MAAiC,0C;K;kDAEjC,e;MAAyD,W;K;mDACzD,8B;MAA4E,c;K;mDAC5E,  
mB;MAAwE,c;K;uDACxE,e;MAA8D,W;K;+CAC9D,Y;MAAsC,Q;K;+CACtC,Y;MAAyC,8B;K;;Iab7C,6C;MA  
AA,4C;QAAA,2B;OAAA,qC;K;IAqB8B,wC;MAC1B,kB;MACA,wB;K;4CAGA,e;MAGQ,Q;MAFJ,UAAU,I;MA  
CV,OAAO,IAAP,C;QACI,YAAA,GAAl,UAAJ,aAAy,GAAZ,W;UAAwB,W;SACxB,WAAW,GAAl,O;QACf,IA  
AI,oCAAJ,C;UACI,MAAM,I;;UAEN,OAAO,iBAAK,GAAL,C;;K;6CAKnB,8B;MACI,iBAAU,WAAK,cAAK,O  
AAL,EAAC,SAAd,CAAF,EAAyC,cAAzC,C;K;iDAEJ,e;UAGW,I;MAFP,+BAAQ,GAAR,U;QAAoB,OAAO,W;O  
AC3B,cAAc,WAAK,kBAAS,GAAT,C;MAEf,gBAAy,WAAZ,C;QAAoB,W;WACpB,gBAAy,mCAAZ,C;QAAq  
C,qB;;QAC7B,2BAAgB,OAAhB,EAAyB,cAAzB,C;MAHZ,W;K;uCAOJ,Y;MAIc,IAAI,IAAJ,Q;MAHV,UAAU,I;  
MACV,WAAW,C;MACX,OAAO,IAAP,C;QACU,uBAAI,OAAJ,GAAl,OAAJ,gC;QAAA,mB;UAAgC,OAAO,I;S  
AA7C,MAAM,M;QACN,mB;;K;2CAIR,mB;MACI,+BAAI,OAAQ,IAAZ,GAAoB,OAAPB,C;K;8CAEJ,mB;MAQ  
4B,Q;MAPxB,UAAU,O;MACV,OAAO,IAAP,C;QACI,IAAI,CAAC,gBAAS,GAAl,UAAb,CAAL,C;UAA4B,OA  
AO,K;QACnC,WAAW,GAAl,O;QACf,IAAI,oCAAJ,C;UACI,MAAM,I;;UAEN,OAAO,gBAAS,0EAAT,C;;K;uC  
AKnB,iB;MACI,gBAAS,KAAT,KAakB,yCAA4B,KAAM,SAAN,KAAGB,aAA5C,IAAsD,KAAM,eAAy,IAAZ,C  
AA9E,C;K;yCAEJ,Y;MAA+B,OAAK,SAAL,WAAK,CAAL,GAA0B,SAAR,cAAQ,CAA1B,I;K;IAGZ,uD;MACX  
,OAAI,G3JyHoC,YAAU,C2JzHID,GAAMB,OAAQ,WAA3B,GAA6C,GAAF,UAAQ,O;K;yCAF3D,Y;MACI,aAA  
M,kBAAK,EAAL,EAAS,+BAAT,CAAN,GAEL,G;K;IAMO,8E;MAAA,6B;QAAyB,Q;QAAT,iBAAS,sBAAT,EA  
AS,8BAAT,UAAoB,O;QAAQ,W;O;K;+CAJ3D,Y;MAOsB,Q;MANIB,QAAQ,a;MACR,eAAe,gBAA+B,CAA/B,O  
;MACf,gBAAy,CAAZ,C;MACA,kBAAK,kBAAL,EAAW,oDAAX,C;M9KtFJ,IAAI,E8KuFM,YAAS,C9KvFf,CA  
AJ,C;QACI,cAdW,e;QAEX,MAAM,6BAAsB,OAAQ,WAA9B,C;O8KuFN,OAAO,+BAAW,qDAAX,C;K;IAGa,8

C;MACpB,kD;MADqB,wB;K;IACrB,gD;MAAA,oD;MACI,4B;K;;;IADJ,4D;MAAA,2D;QAAA,0C;OAAA,oD;K; yDAIA,Y;MAA0C,gBAAT,a;M5Lm9YrB,Q;MADhB,kb4Ll9YmD,mC;M5Lm9YnD,wBAAgB,SAAhB,gB;QAAg B,cAAA,SAAhB,M;QAAsB,cAAwB,yBAAa,OAAb,C;;M4Ln9YT,O5Lo9Y9B,W;K;;;I6LtoZX,oE;MA4BI,MAA M,wBAAoB,sEAApB,C;K;8GA5BV,yB;MAAA,2D;MAAA,sC;QA4BI,MAAM,6BAAoB,sEAApB,C;O;KA5BV, C;IA0CoC,mC;MAAQ,4D;K;IAE5C,4C;MAAA,e;MAAA,iB;MAAA,uB;K;IAAA,0C;MAAA,6C;O;MAK0C,oG; MAAqB,gF;MAAW,4E;K;;IAAhC,+D;MAAA,gC;MAAA,uD;K;;IAAqB,qD;MAAA,gC;MAAA,6C;K;;IAAW,m D;MAAA,gC;MAAA,2C;K;;IAL1E,sC;MAAA,sJ;K;;IAAA,2C;MAAA,a;aaaa,qB;UAAA,4D;aAAA,W;UAAA,k D;aAAA,S;UAAA,gD;gBAAA,qF;;K;;6ECnDA,yB;MAAA,0B;MAAA,mC;QAGsD,OAAiC,OAA3B,SAAL,GAA uB,KAAS,C;O;KAHvF,C;2EAKA,yB;MAAA,0B;MAAA,mC;QAGqD,OAAgC,OAA1B,SAAL,GAAsB,KAAS,C; O;KAHrF,C;6EAKA,yB;MAAA,0B;MAAA,mC;QAGsD,OAAiC,OAA3B,SAAL,GAAuB,KAAS,C;O;KAHvF,C;6 EAKA,yB;MAAA,0B;MAAA,4B;QAGqC,OAAqB,OAAP,CAAR,SAAE,C;O;KAH1D,C;+EAMA,yB;MAAA,4B; MAAA,mC;QAGyD,OAAiC,QAA3B,SAAL,GAAuB,KAAS,C;O;KAH1F,C;6EAKA,yB;MAAA,4B;MAAA,mC; QAGwD,OAAgC,QAA1B,SAAL,GAAsB,KAAS,C;O;KAHxH,C;+EAKA,yB;MAAA,4B;MAAA,mC;QAGyD,OA AiC,QAA3B,SAAL,GAAuB,KAAS,C;O;KAH1F,C;+EAKA,yB;MAAA,4B;MAAA,4B;QAGuC,OAAqB,QAAP,C AAR,SAAE,C;O;KAH5D,C;ICpCA,qC;K;;ICAA,mB;K;;IAOA,iB;K;;IAOA,2C;K;;IAOA,wB;K;;IAQA,0B;K;;IA OA,sB;K;;IAOA,4B;K;;IAOA,6C;K;;IA+BuC,wE;MAEnC,uB;QAAA,UAAAsB,E;MACtB,qB;QAAA,8B;MACA,2 B;QAAA,qE;MACA,yB;QAAA,YAAqB,E;MAJrB,sB;MACA,sB;MACA,kB;MACA,8B;MACA,0B;K;;IAGJ,iD; MAAA,e;MAAA,iB;MAAA,uB;K;IAAA,+C;MAAA,kD;O;MAKI,wG;MACA,wG;MACA,8F;K;;IAFA,iE;MAAA ,qC;MAAA,yD;K;;IACA,iE;MAAA,qC;MAAA,yD;K;;IACA,4D;MAAA,qC;MAAA,oD;K;;IAPJ,2C;MAAA,6K;K ;;IAAA,gD;MAAA,a;aaaa,kB;UAAA,8D;aAAA,kB;UAAA,8D;aAAA,a;UAAA,yD;gBAAA,6E;;K;;IAUA,wB;K ;;ICjGA,qB;MAAA,yB;K;0CAII,Y;MAO6D,uB;K;2HAE7D,yB;MAAA,+D;MAAA,kC;MAAA,0F;MAAA,6F;MA AA,4E;QAUI,wC;QAAS,2C;O;MAVb,mEAWQ,wC;QAA6E,sBAAS,QAAT,EAAMB,QAANB,EAA6B,QAA7B,C ;O;MAXrF,oG;MAAA,yC;QAUI,wDAA+B,YAA/B,C;O;KAVJ,C;uHAcA,yB;MAAA,+D;MAAA,kC;MAAA,wF; MAAA,yF;MAAA,0E;QAcI,wC;QAAS,2C;O;MADB,kEAeQ,wC;QAAuF,6BAAS,QAAT,EAAMB,QAANB,EAA6 B,QAA7B,C;O;MAf/F,kG;MAAA,yC;QAcI,sDAA+B,YAA/B,C;O;KAdJ,C;;;IA3BJ,iC;MAAA,gC;QAAA,e;OAA A,yB;K;IAgDiC,sB;MAC7B,eAAwB,I;K;4CAExB,6B;MACW,Q;MAAA,mB;MAAA,iB;QAAS,MAAM,6BAAsB ,cAAY,QAAS,aAArB,uCAAtB,C;OAAtB,OAAO,I;K;4CAGX,oC;MACI,eAAa,K;K;;;kDC9CjB,6B;;K;;;iEA+ CA,6B;;K;;ICrDuC,0C;MACvC,uBAAoB,Y;K;wDAEpB,wC;MAM6F,W;K;uDAE7F,wC;K;oDAMA,6B;MACI,O AAO,oB;K;oDAGX,oC;MACI,eAAe,IAAK,gB;MACpB,IAAI,CAAC,0BAAa,QAAb,EAAuB,QAAvB,EAAiC,KA AjC,CAAL,C;QACI,M;OAEJ,uBAAa,K;MACb,yBAAy,QAAsB,EAAsB,QAAtB,EAAgC,KAAhC,C;K;;4EC9BR, wC;MAqBI,OAAO,e;K;4EAGX,+C;MAuBI,cAAI,KAAJ,C;K;4EAIJ,wC;MAMBI,OAAO,cAAI,OAAJ,C;K;4EAG X,+C;MAqBI,cAAI,OAAJ,EAAa,KAAb,C;K;IC/FJ,kB;MA6PI,4B;K;+BAtoA,Y;MAOiC,6BAAS,EAAT,C;K;uC AEjC,iB;MAW2C,4BAAQ,CAAR,EAAW,KAAx,C;K;uCAE3C,uB;MAAkB,Q;MAHd,iBAAiB,IAAjB,EAAuB,K AAavB,C;MACA,QAAQ,QAAQ,IAAR,I;MACR,IAAI,IAAI,CAAJ,IAAS,MAAK,WAAIB,C;QACc,IAAI,MAAM, CAAC,CAAD,IAAN,OAAY,CAAhB,C;UACN,eAAe,SAAS,CAAT,C;UACf,6BAAS,QAAT,C;;UAEA,K;;YAEI, WAAW,cAAU,KAAK,C;YAC1B,IAAI,OAAO,C;;UACN,gBAAO,CAAP,IAAY,CAAZ,GAAgB,CAAhB,SAAqB, CAArB,C;UACT,Q;;QATJ,c;QAWA,OAAO,OAAO,GAAP,I;;QAEP,OAAO,IAAP,C;UACI,YAAU,c;UACV,IAA W,IAAP,qBAAkB,KAAtB,C;YAA6B,OAAO,K;;K;gCAKhD,Y;MAOmC,OAAU,oBAAV,cAAU,CAAS,WAAI,E AAJ,CAANB,yBAA6B,cAA7B,E;K;wCAEnC,iB;MAW8C,iCAAY,KAAZ,C;K;wCAE9C,uB;MAiBkB,Q;MAPd,m BAAiB,IAAjB,EAAuB,KAAvB,C;MACA,QAAQ,eAAQ,IAAR,C;MACR,IAAI,eAAI,CAAR,C;QACI,O;QACA,I AAI,aAAO,CAAD,aAAN,GAAY,CAAZ,CAAJ,C;UACI,WAAW,CAAE,Q;UACb,YAAa,qBAAO,EAAP,CAAW, Q;UAEpB,aAAQ,CAAR,C;YACI,eAAe,SAAS,IAAT,C;YAEf,OAAmB,oBAAAnB,sBAAS,QAAT,CAAmB,CAAn B,iB;iBAEJ,cAAS,CAAT,C;YAEI,OAAU,oBAAV,cAAU,CAAV,iB;;YAEA,iBAAe,SAAS,KAAT,C;YACf,OAA mB,oBAAAnB,sBAAS,UAAT,CAAmB,CAAS,WAAI,EAAJ,CAA5B,KAAiD,oBAAV,cAAU,CAAV,iBAAvC,C;;U AXR,U;;UAeA,K;;YAEI,WAAW,eAAW,oBAAK,CAAL,C;YACtB,IAAI,YAAO,CAAP,C;;UACC,sBAAO,CAAP ,MAAY,+BAAI,CAAJ,EAAZ,eAAqB,CAArB,C;UACT,MAAM,C;;QAEV,OAAO,SAAO,GAAP,C;;QAEP,OAA O,IAAP,C;UACI,YAAU,e;UACV,IAAW,IAAP,0CAAkB,KAAIB,CAAJ,C;YAA6B,OAAO,K;;K;mCAKhD,Y;M AKyC,6BAAS,CAAT,MAAE,C;K;kCAExD,Y;MAKuC,uBAAgB,sBAAS,EAAT,CAAhB,EAA8B,sBAAS,EAAT,

CAA9B,C;K;0CAEvC,iB;MASoD,+BAAW,GAAX,EAAgB,KAAhB,C;K;0CAEpD,uB;MAcY,Q;MAFR,mBAAiB,IAAjB,EAAuB,KAAvB,C;MACA,WAAW,QAAQ,I;MACX,IAAS,WAAI,IAAK,CAAL,IAA0B,SAAL,IAAK,CAA1B,IAA8C,SAAN,KAAM,CAAD,C;QACJ,SAAS,qBAAgB,QAAQ,CAAR,GAAY,OAAO,CAAnc,C;QACT,cAAO,EAAP,GAAY,E;;QAEZ,cAAO,oBAaE,I;;MAJ1B,Y;MAMA,OAAW,KAAK,KAAT,GAAsB,SAAN,KAAM,CAAAtB,GAAsC,C;K;iCAGjD,Y;MAKqC,6BAAS,EAAT,IAA0B,Q;K;IAWK,oF;MAAA,mB;QAAE,uBAAa,iBAAB,sBAAqC,eAArC,+BAAqE,aAAM,OAA3E,M;O;K;iDATtE,qC;MvLjLA,IAAI,EUl0LqB,CAAb,8BAAgB,KAAM,OvL1L9B,GuL0LiD,CAAX,0BAAc,KAAM,OvL1L1D,GuL0LsC,KvL1LtC,CAAJ,C;QACI,cuLyLgE,kDvLzLID,E;QACd,MAAM,gCAAyB,OAAQ,WAAjC,C;OAFV,IAAI,EUl2LQ,aAAa,OvL3LrB,CAAJ,C;QACI,gBuL0LgC,mF;QvLzLhC,MAAM,gCAAyB,SAAQ,WAAjC,C;OuL2LN,YAAY,CAAC,UAAU,SAAV,IAAD,IAAwB,CAAxB,I;MAEZ,mBAaE,SAAf,C;MpLzEJ,iBAAc,CAAd,UoL0EW,KpL1EX,U;QoL2EQ,QAAQ,c;QACR,MAAM,UAAU,IAAoB,OAAf,CAAE,C;QACpB,MAAM,aAAW,CAAX,IAAN,IAAgC,OAAV,CAAE,KAAK,CAAG,C;QACqC,MAAM,aAAW,CAAX,IAAN,IAAiC,OAAZ,I;MAGJ,gBAAgB,UAAU,UAAV,I;MACHB,SAAS,sBAAS,YAAY,CAAZ,IAAT,C;MACT,aAAU,CAAV,MAAkB,SAAlB,M;QACI,MAAM,aAAW,CAAX,IAAN,IAAqC,OAaf,EAAG,MAAK,IAAI,CAAJ,IAAL,CAAY,C;;MAGzC,OAAO,K;K;yCACX,uD;MAvB4C,yB;QAAA,YAAiB,C;MAAG,uB;QAAA,UAAe,KAAM,O;aArF,0H;K;yCAiCA,iB;MAOyD,8BAAU,KAAV,EAAiB,CAAjB,EAAoB,KAAM,OAA1B,C;K;yCAEzD,gB;MAKkD,8BAAU,cAAU,IAAV,CAAV,C;K;IAGiD,0B;MAAA,8B;MAO2B,iB;MACvB,uBAAoC,uB;K;IAEpC,qC;MAAA,yC;MACI,4B;K;wDAEA,Y;MAAiC,mC;K;;IAHrC,iD;MAAA,gD;QAAA,+B;OAAA,yC;K;8CAMA,Y;MAAkC,8C;K;gDAEIC,oB;MAA4C,OAAA,oBAAc,kBAAS,QAAT,C;K;uCAC1D,Y;MAA8B,OAAA,oBAAc,U;K;+CAC5C,iB;MAAwC,OAAA,oBAAc,iBAAQ,KAAR,C;K;+CACtD,uB;MAAmD,OAAA,oBAAc,iBAAQ,IAAR,EAAC,KAAd,C;K;wCAEjE,Y;MAAgC,OAAA,oBAAc,W;K;gDAC9C,iB;MAA2C,OAAA,oBAAc,kBAAS,KAAT,C;K;gDACzD,uB;MAAuD,OAAA,oBAAc,kBAAS,IAAT,EAAe,KAAf,C;K;2CAErE,Y;MAAsC,OAAA,oBAAc,c;K;0CAEpD,Y;MAAoC,OAAA,oBAAc,a;K;kDACID,iB;MAAiD,OAAA,oBAAc,oBAAW,KAAX,C;K;kDAC/D,uB;MAA+D,OAAA,oBAAc,oBAAW,IAAX,EAAiB,KAAjB,C;K;yCAE7E,Y;MAAkC,OAAA,oBAAc,Y;K;iDAEhD,iB;MAAsD,OAAA,oBAAc,mBAAU,KAAV,C;K;iDACpE,gB;MAA+C,OAAA,oBAAc,mBAAU,IAAV,C;K;yDAC7D,qC;MACI,OAAA,oBAAc,mBAAU,KAAV,EAAiB,SAAjB,EAA4B,OAA5B,C;K;;IAtCtB,sC;MAAA,qC;QAAA,oB;OAAA,8B;K;;IA0CJ,wB;MAAuC,yBAAa,IAAb,EAAMB,IAAK,IAALEAA5B,C;K;IAEvC,wB;MAAwC,yBAAa,IAAK,QAAlB,EAA2B,IAAK,YAAl,EAJ,CAAQ,QAAX,C;K;IAGxC,mC;MAUI,IAAA,KAAM,UAAU,C;QAAMB,MAAM,gCAAyB,uCAAoC,KAA7D,C;WACzB,IAAA,KAAM,KAAAN,GAAa,UAAb,C;QAF8C,OAEhB,0BAAQ,KAAM,MAAd,EAAqB,KAAM,KAAAN,GAAa,CAAb,IAArB,C;WAC9B,IAAA,KAAM,MAAN,GAAC,WAAAd,C;QAH8C,OAGf,0BAAQ,KAAM,MAAN,GAAC,CAAd,IAAR,EAAYB,KAAM,KAA/B,IAAuC,CAAvC,I;;QAHe,OAIrC,mB;K;IAGZ,oC;MAUI,IAAA,KAAM,UAAU,C;QAAMB,MAAM,gCAAyB,uCAAoC,KAA7D,C;WACzB,IAAA,KAAM,KAAAN,+C;QAFiD,OAEiB,2BAAS,KAAM,MAAf,EAAsB,KAAM,KAAAN,yBAAa,CAAb,EAAtB,C;WAC/B,IAAA,KAAM,MAAN,+C;QAHiD,OAGjB,2BAAS,KAAM,MAAN,8BAAC,CAAd,EAAT,EAA0B,KAAM,KAAhC,0BAAwC,CAAxC,E;;QAHiB,OAIzC,oB;K;IAOZ,yB;MAAYC,YjFrTkB,MAAO,OiFqTpB,KjFrToB,CiFqTzB,I;K;IAEzC,4C;MAEI,OAAA,SAAK,KAAK,EAAL,GAAU,QAAlf,GAAYC,CAAX,CAAC,QAAD,IAAW,KAAI,E;K;IAEjD,uC;MvLtVI,IAAI,EUlSvUD,QAAQ,IvLtV/D,CAAJ,C;QACI,cuLqVuE,+B;QvLpVvE,MAAM,gCAAyB,OAAQ,WAAjC,C;Q;IuLqVd,yC;MvLvVI,IAAI,EUlUVyD,sBAAQ,IAAR,KvLvVzD,CAAJ,C;QACI,cuLsVyE,+B;QvLrVzE,MAAM,gCAAyB,OAAQ,WAAjC,C;Q;IuLsVd,yC;MvLxVI,IAAI,EUlWV6D,QAAQ,IvLxVrE,CAAJ,C;QACI,cuLuV6E,+B;QvLtV7E,MAAM,gCAAyB,OAAQ,WAAjC,C;Q;IuLwVd,yC;MAAYD,oCAA0B,IAA1B,qBAAiC,KAAjC,kB;K;ICrXzD,6B;MAOqC,OnMmYE,SmMnYF,mBnMmYE,C;K;ImMjYvC,sC;MASgD,6BAAS,WAAW,EAAs,KAAb,C;K;IAEhD,4C;MAUI,qBAAqB,IAArB,EAA2B,KAA3B,C;MAEA,iBAAiB,InMqQgB,KmMrQhB,GAAiB,W;MACiC,kBAAkB,KnMoQe,KmMpQf,GAAB,W;MAEpC,mBAAMB,0BAAQ,UAAU,EAAs,WAApB,IAAqC,W;MACxD,OnMsWmC,SmMtW5B,YnMsW4B,C;K;ImMnWvC,sC;MAWI,IAAA,KAAM,UAAU,C;QAAMB,MAAM,gCAAyB,uCAAoC,KAA7D,C;;QACzB,InMGkE,YmMHIE,KAAM,KnMG6E,KAAjB,EmMHRd,4BAAK,UnMG6E,KAA7B,CmMHIE,K;UAFiD,OAEiB,sBAAS,KAAM,MAAf,EnMqBsB,SmMrBA,KAAM,KnMqBI,KAAK,GAAW,CmMrBb,WnMqBa,MAAX,IAAf,CmMrBtB,C;;UAC/B,InMEkE,YmMFIE,KAAM,MnME6E,KAAjB,EmMFpD,4BAAK,UnME4E,KAA7B,CmMFIE,K;YA

HiD,OnMuBI,SmMpBrB,sBnMiCsB,SmMjCb,KAAM,MnMiCiB,KAAK,GAAY,CmMjC1B,WnMiC0B,MAAZ,IA Af,CmMjCtB,EAA2B,KAAM,KAAjC,CnMoB+B,KAAK,GAAW,CmMpBN,WnMoBM,MAAX,IAAf,C;;YmMvB J,OAIzC,mB;;;K;IAGZ,8B;MAOuC,OnL0VG,Uml1VH,oBnL0VG,C;K;ImLxV1C,uC;MASmD,8BAAU,2BAAV, EAAe,KAAf,C;K;IAEnD,6C;MAUI,sBAAsB,IAAtB,EAA4B,KAA5B,C;MAEA,iBAAiB,InLwNkB,KmLxN1B,8B; MACjB,kBAAkB,KnLuNiB,KmLvNjB,8B;MAEiB,mBAAmB,2BAAS,UAAT,EAAqB,WAArB,+B;MACnB,OnL 6TsC,Uml7T/B,YnL6T+B,C;K;ImL1T1C,uC;MAWI,IAAA,KAAM,UAAN,C;QAAMB,MAAM,gCAAYB,uCAAo C,KAA7D,C;;QACzB,InL7CmE,amL6CnE,KAAM,KnL7C+E,KAAiB,EmL6CtD,6BAAM,UnL7C8E,KAA9B,Cm L6CnE,K;UAFoD,OAEPB,uBAAU,KAAM,MAAhB,EnLhCuB,UmlGcA,KAAM,KnLhCK,KAAK,KAAW,ChBs Q7C,UAAW,oBAAL,CmMtOyB,WnMsOzB,MAAK,CAAL,iBAAN,CgBtQ6C,MAAX,CAAhB,CmLgCvB,C;;UA ChC,InL9CmE,amL8CnE,KAAM,MnL9C+E,KAAiB,EmL8CrD,6BAAM,UnL9C6E,KAA9B,CmL8CnE,K;YAHo D,OnL9BG,UmlLiCtB,uBnLpBuB,UmlObB,KAAM,MnLpBkB,KAAK,UAAY,ChByP/C,UAAW,oBAAL,CmMrO c,WnMqOd,MAAK,CAAL,iBAAN,CgBzP+C,MAAZ,CAAhB,CmLoBvB,EAA4B,KAAM,KAAiC,CnLjCiC,KAA K,KAAW,ChBsQ7C,UAAW,oBAAL,CmMrOgC,WnMqOhC,MAAK,CAAL,iBAAN,CgBtQ6C,MAAX,CAAhB,C ;;YmL8BH,OAI5C,oB;;;K;IAGZ,sC;MAQI,4BAAU,KhK4+FH,QgK5+FP,C;MACA,OAAO,K;K;IAGX,uC;MAKs D,OhK2iG3C,egK3iG2C,4BAAU,IAAV,ChK2iG3C,C;K;IgKziGX,4D;MAOGD,yB;QAAA,YAAiB,C;MAAG,uB; QAAA,UAAe,KAAM,K;MACrF,4BAAU,KhKy9FH,QgKz9FP,EAA+B,SAA/B,EAA0C,OAA1C,C;MACA,OAA O,K;K;IAIX,2C;MxLrHI,IAAI,EX2B8D,YmM0FD,KnM1FkB,KAAjB,EmM0FO,InM1FsB,KAA7B,CmM0FD,Ix LrH7D,CAAJ,C;QACI,cwLoH6E,+B;QxLnH7E,MAAM,gCAAYB,OAAQ,WAAjC,C;Q;IwLoHd,4C;MxLrHI,IAAI ,EKmC+D,amLmFC,KnLnFiB,KAAiB,EmLmFS,InLnFqB,KAA9B,CmLmFC,IxLrHhE,CAAJ,C;QACI,cwLqHgF, +B;QxLpHhF,MAAM,gCAAYB,OAAQ,WAAjC,C;Q;IyLpBc,6C;MASCxB,oC;MA/BA,iB;MANA,Y;MACA,Y;M ACA,Y;MACA,Y;MACA,Y;MACA,sB;MzLYA,IAAI,EyLLQ,CAAC,WAAK,QAAL,GAAU,QAAY,GAAe,QAaf ,GAAoB,QAARB,MAA2B,CzLKnC,CAAJ,C;QACI,cyLNwC,wD;QzLOxC,MAAM,gCAAYB,OAAQ,WAAjC,C;O GoHV,iBAAc,CAAd,UsLxHW,EtLwHX,U;QsLxHiB,c;;K;qCAGjB,Y;MAGI,QAAQ,Q;MACR,IAAI,IAAO,MAA O,C;MACIB,WAAI,Q;MACJ,WAAI,Q;MACJ,WAAI,Q;MACJ,SAAS,Q;MACT,WAAI,E;MACJ,IAAK,IAAO,KA AM,CAAd,GAAsB,EAAtB,GAA8B,MAAO,C;MACzC,WAAI,C;MACJ,gCAAU,MAAV,I;MACA,OAAO,IAAI,a AAJ,I;K;8CAGX,oB;MACI,OAAU,cAAV,cAAU,EAAc,QAAd,C;K;IAEd,kC;MAAA,sC;MACI,4B;K;;IADJ,8C; MAAA,6C;QAAA,4B;OAAA,sC;K;;IA7BA,gD;MAAA,sD;MACQ,yBAAK,KAAL,EAA Y,KAAZ,EAAmB,CAAn B,EAAAsB,CAAtB,EAA+B,CAAN,KAAzB,EAAuC,SAAU,EAA X,GAAoB,UAAW,CAArE,C;MADR,Y;K;ICbiD, 8C;MACjD,4B;MACA,0C;K;oEADA,Y;MAAA,2B;K;2EACA,Y;MAAA,kC;K;uCAGA,iB;MACI,OAAO,0CAA g C,kBAaA,KAAM,UAAAnB,KAC/B,mBAAS,KAAM,MAAf,KAAwB,0BAAgB,KAAM,aAAtB,CADO,CAAhC,C; K;yCAIX,Y;MACI,OAAW,cAAJ,GAAe,EAAf,GAAuB,MAAW,SAAN,UAAAM,CAAX,QAAqC,SAAb,iBAaA,CA ArC,I;K;yCAGIC,Y;MAAkC,OAAE,UAAf,qBAAU,iB;K;;IAGhD,kC;MAM6E,2BAAgB,SAAhB,EAAAsB,IAAtB, C;K;;;0DAYzE,iB;MAA2C,qCAAiB,UAAjB,EAAwB,KAAxB,KAAkC,8BAAiB,KAAjB,EAAwB,iBAAXB,C;K;i DAC7E,Y;MAAkC,QAAC,8BAAiB,UAAjB,EAAwB,iBAAXB,C;K;;IAcR,gD;MAI3B,gBAAqB,K;MACrB,uBAA 4B,Y;K;0FACD,Y;MAAQ,oB;K;iGACD,Y;MAAQ,2B;K;2DAE1C,gB;MAA+D,YAAK,C;K;mDAEPe,iB;MAAg D,gBAAS,aAAT,IAAmB,SAAS,oB;K;0CAC5E,Y;MAAkC,SAAE,iBAAU,oBAAZ,C;K;yCAEIC,iB;MACI,OAA O,4CAA+B,kBAaA,KAAM,UAAAnB,KAC9B,kBAAU,KAAM,SAAhB,IAA0B,yBAAiB,KAAM,gBADnB,CAA/B, C;K;2CAIX,Y;MACI,OAAW,cAAJ,GAAe,EAAf,GAAuB,MAAY,SAAP,aAAO,CAAZ,QAAuC,SAAd,oBAAc,C AAvc,I;K;2CAGIC,Y;MAAkC,OAAE,aAAF,qBAAW,oB;K;;IAGjD,oC;MAOqF,6BAAkB,SAAIB,EAAwB,IAAx B,C;K;IAQvD,+C;MAI1B,gBAAqB,K;MACrB,uBAA4B,Y;K;yFACF,Y;MAAQ,oB;K;gGACD,Y;MAAQ,2B;K;0 DAEzC,gB;MAA6D,YAAK,C;K;kDAEIE,iB;MAA+C,gBAAS,aAAT,IAAmB,SAAS,oB;K;yCAC3E,Y;MAAkC,S AAE,iBAAU,oBAAZ,C;K;wCAEIC,iB;MACI,OAAO,2CAA8B,kBAaA,KAAM,UAAAnB,KAC7B,kBAAU,KAAM ,SAAhB,IAA0B,yBAAiB,KAAM,gBADpB,CAA9B,C;K;0CAIX,Y;MACI,OAAW,cAAJ,GAAe,EAAf,GAAuB,M AAY,SAAP,aAAO,CAAZ,QAAuC,SAAd,oBAAc,CAAvc,I;K;0CAGIC,Y;MAAkC,OAAE,aAAF,qBAAW,oB;K;; IAGjD,oC;MAOkF,4BAAiB,SAAjB,EAAuB,IAAvB,C;K;oFAGIF,8B;MAQI,0BAAmB,2BAAS,OAAT,C;K;IAGv B,+C;MACI,IAAI,CAAC,UAAAL,C;QAAiB,MAAM,gCAAYB,iCAA8B,IAA9B,iBAAzB,C;K;IC5I3B,gC;MACw, Q;MADP,IAAI,CAAC,6BAAW,KAA X,CAAL,C;QAAwB,MAAM,uBAAmB,sC/EjBzC,oB+EiByC,CAAnB,C;O AC9B,OAAO,sD;K;IAMX,oC;MAAkC,Q;MAA9B,OAAW,6BAAW,KAA X,CAAJ,GAAuB,sDAAvB,GAAuC,I;K;

;;;;;ICvBhB,yC;MA2B9B,uC;MA1BA,wB;MAIA,gB;M5LQA,IAAI,E4LDS,iBAAY,IAAb,MAAuB,iBAAvB,C5L  
CR,CAAJ,C;QACI,c4LDQ,iBAAY,IAAhB,GACI,8CADJ,GAGI,sCAA0B,aAA1B,qC;Q5LDR,MAAM,gCAAYB,  
OAAQ,WAAjC,C;Q;yC4LKV,Y;MAAwC,Q;MAAA,oB;MACpC,iB;QAD8B,OACtB,G;WACR,oD;QAF8B,OAE  
F,SAAL,SAAK,C;WAC5B,6C;QAH8B,OAGd,iBAAK,SAAL,C;WACbB,8C;QAJ8B,OAIb,kBAAM,SAAN,C;;Q  
AJa,mC;K;IAOIC,qC;MAAA,yC;MACI,YAGqC,oBAAgB,IAAhB,EAA sB,IAAtB,C;K;iGAQJ,Y;MAAQ,gB;K;4D  
AEzC,gB;MAOI,8DAAqC,IAArC,C;K;gEAEJ,gB;MAMI,uDAA8B,IAA9B,C;K;4DAEJ,gB;MAMI,wDAA+B,IA  
A/B,C;K;;IArCR,iD;MAAA,gD;QAAA,+B;OAAA,yC;K;;2CArCJ,Y;MAWI,oB;K;2CAXJ,Y;MAeI,gB;K;6CAfJ,0  
B;MAAA,2BAWI,8CAXJ,EAeI,kCAfJ,C;K;yCAA,Y;MAAA,c;MAWI,yD;MAIA,qD;MAfJ,a;K;uCAA,iB;MA  
AA,4IAWI,4CAXJ,IAeI,oCAfJ,I;K;ICLA,kC;MAAA,e;MAAA,iB;MAAA,uB;K;IAAA,gC;MAAA,mC;O;MAYI,4  
D;MAKA,8C;MAKA,gD;K;;IAVA,2C;MAAA,sB;MAAA,mC;K;;IAKA,oC;MAAA,sB;MAAA,4B;K;;IAKA,qC;  
MAAA,sB;MAAA,6B;K;;IAtBJ,4B;MAAA,mG;K;;IAAA,iC;MAAA,a;AAAA,W;UAAA,wC;aAAA,I;UAAA,iC;a  
AAA,K;UAAA,kC;gBAAA,6D;;K;;6ECAA,yB;MAAA,4F;MAAA,2B;QASI,MAAM,mCAA8B,0EAA9B,C;O;KA  
TV,C;ICkCA,+D;MAaW,Q;MAAP,OAAO,8CAA0,KAAP,EAAC,UAAAd,EAA0B,QAA1B,oC;K;IAGX,kC;MAIiB  
,Q;MAAb,wBAAa,KAAb,gB;QAAa,WAAA,KAAb,M;QACI,yBAAO,IAAP,C;;MACJ,OAAO,S;K;mFAGX,qB;M  
AGwD,gCAA0,EAAP,C;K;qFAExD,4B;MAG4E,OAAA,yBAAO,KAAP,CALpB,gBAAO,EAAP,C;K;qFAOxD,4  
B;MAGmE,OAAA,yBAAO,KAAP,CAVX,gBAAO,EAAP,C;K;IAxD,wD;MAEQ,sB;QAAqB,yBAAO,UAAU,O  
AAV,CAAP,C;WACrB,sD;QAA4B,yBAAO,OAAP,C;WAC5B,2B;QAAmB,yBAAO,kBAAP,C;;QACX,yBAAe,S  
AAR,OAAQ,CAAF,C;K;IjL7EhB,+B;MAY6B,kBAAIB,QAAQ,SAAR,EAAC,EAAd,C;MACH,IX0EE,WW1EE,G  
AAK,CAAT,C;QAAy,MAAM,gCAAYB,oEAAzB,C;MADtB,OX4EO,W;K;IWvEX,wC;MAGBW,Q;MAAA,qCA  
AiB,KAAjB,C;MAAA,iB;QAA2B,MAAM,gCAAYB,8BAAO,SAAP,4CAA+C,KAAxE,C;OAAxC,OAAO,I;K;IA  
GX,qC;MAY6B,kBAAIB,QAAQ,SAAR,EAAC,EAAd,C;MAAP,OXmEqB,WWnEa,IAAM,CXmEjC,GAAqB,WA  
ArB,GAA+B,I;K;IWhE1C,8C;MAGBI,WAAW,KAAAX,C;MAC4B,kBAArB,QAAQ,SAAR,EAAC,KAAd,C;MAAP  
,OX+CqB,WW/CgB,IAAM,CX+CpC,GAAqB,WAArB,GAA+B,I;K;IW5C1C,gC;MAWI,IAAY,CAAR,8BAAW,C  
AAf,C;QACI,OAAO,YAAM,SAAN,C;OAEX,MAAM,gCAAYB,SAAM,SAAN,4BAAzB,C;K;IAGV,yC;MAKBW,  
Q;MANP,IAAI,EAAU,CAAV,sBAAa,EAAb,CAAJ,C;QACI,MAAM,gCAAYB,oBAAiB,KAAjB,4CAAzB,C;OAE  
V,IAAI,YAAO,CAAP,IAAY,aAAQ,KAAxB,C;QACI,MAAM,gCAAYB,WAAQ,SAAR,mDAawD,KAAjF,C;OA  
EH,IAAI,YAAO,EAAX,C;QACH,mBAAM,SAAN,C;;QAEA,0BAAM,SAAN,IAAa,EAAb,C;;MAHJ,W;K;IAuFJ,  
8B;MAWsC,+B;K;0EAEtC,4B;MAM8D,OAAK,oBAAL,SAAK,CAAL,GAAB,K;K;IAEHF,gD;MAQoC,0B;QA  
AA,aAAsB,K;MACtD,IAAI,cAAQ,KAAZ,C;QAAmB,OAAO,I;MAC1B,IAAI,CAAC,UAAL,C;QAAiB,OAAO,K;  
MAExB,gBAAqB,cAAL,SAAK,C;MACrB,iBAAuB,cAAN,KAAM,C;MAEHb,yBAAa,U;MAAb,U;QAA2B,OfR  
MyB,oBEqMzB,SFrMyB,CAAY,cAfrB,YAAY,CAAZ,CEoNhB,KFrMyB,oBEqMI,UFrMJ,CAAY,cAfrB,YAAY,  
CAAZ,C;OEO NID,W;K;IAGJ,gC;MAGyC,QAAQ,cAAA,sCAAK,cAAL,EAAoB,sCAAK,cAAzB,CAAR,6B;K;Ik  
L3OzC,6C;MAc6B,4B;QAAA,eAAuB,G;MACHD,wCAAsB,EAAtB,EAA0B,YAA1B,C;K;IAEJ,mE;MAKwC,yB;  
QAAA,YAAoB,E;MAAI,4B;QAAA,eAAuB,G;MhMGnF,IAAI,CmBwR+C,CAAC,Q6K1R5C,Y7K0R4C,CnBxRp  
D,C;QACI,cgMHiC,wC;QhMIjC,MAAM,gCAAYB,OAAQ,WAAjC,C;OgMHV,cAAY,gB;MAEC,yBAAS,mBAA  
S,YAAA,SAAU,OAAV,EAAMB,OAAM,KAAzB,CAAT,I;MAAT,wBAAiD,kBAAB,SAAlB,C;MA0E9D,gBAA  
gB,iBA1ET,OA0ES,C;M1Lg7CT,kBAAoB,gB;MAoSd,gB;MADb,YAAY,C;MACC,00L9xDN,01L8xDM,W;kB  
AAb,OAAa,cAAb,C;QAAa,sB;QA1RsB,U;QAAA,cA0RT,oBAAMB,cAAnB,EAAMB,sBAAnB,U;Q0L/sDIB,kB;;  
YAHA,CAAC,YAAS,CAAT,IAAc,qBAAf,KAA4C,Q1LktDG,I0LltDH,C;UAC5C,a;;UAEA,4B;UA9E+B,uB;;Y9  
KgHzB,kC;YAAA,wBZ6qDyC,IY7qDzC,C;YAAA,qB;YAAA,oB;YAAA,oB;YAAAd,gE;cACI,I8KjHkD,CAAI,aA  
AH,U9KiHrC,YZ4qDqC,IY5qDrC,YAAK,OAAL,E8KjHqC,CAAG,C9KiHtD,C;gBACI,sBAAO,O;gBAAP,wB;;Y  
AGR,sBAAO,E;;U8KrHH,iD;UAGI,gCAA2B,EAA3B,C;YAHJ,2BAGqC,I;IBACjC,IAAK,a1LyxD0C,I0LzxD1C,  
gBAAyB,uBAAzB,CAAL,C;YAJJ,2B1L6xDmD,IOjmDsB,WmLxLI,0BAAuC,mBAAvC,InLwLJ,C;;YmL5LzE,2  
BAKY,I;;UAYER,iE7LJD,yB6LIC,4B1L+sD+C,I;;QA1RpB,8B;UAA6C,6B;;M0LpgDhF,OAIkF,S1Lo7CE,W0Lp  
7CF,EAAO,mBAAc,kBAAd,CAAP,EAA0C,IAA1C,CACA,W;K;IAvET,+B;MAeyC,gCAAc,EAAd,C;K;IAEzC,6  
C;MAGgC,yB;QAAA,YAAoB,E;MAM3C,Q;MALL,cAAY,gB;M1LurBL,kBAAS,gB;MA2FA,U;MAAA,S0LhxB  
M,O1LgxBN,W;MAAhB,OAAGB,gBAAhB,C;QAAGB,2B;QAAM,Ia3hB6B,CAAC,Qb2hBhB,Oa3hBgB,Cb2hB9  
B,C;UAAwB,WAAy,WAAI,OAAJ,C;;M0L9wBrD,kB1L+wBE,W;MAMrBA,oBAAM,iBAAa,qCAAwB,EAAXB,

CAAb,C;MAuEA,U;MAAA,+B;MAAb,OAAa,gBAAb,C;QAAa,wB;QACT,aAAY,uBAAc,IAAd,E;;M0L5gDhB,s  
BAAsB,CAGjB,oB1L0gDE,a0L1gDF,CAHiB,mBAGf,C;MAEP,yBAAS,mBAAS,YAAA,SAAU,OAAV,EAAmB,  
OAMM,KAAzB,CAAT,I;MAAT,wBAAiD,kBAAkB,SAAIB,C;MAMc9D,gBAAGb,iBAnCT,OAmCS,C;M1Lg7C  
T,oBAAoB,gB;MAoSd,kB;MADb,YAAY,C;MACC,S0LvDN,O1LuvDM,W;MAAb,OAAa,gBAAb,C;QAAa,0B;  
QA1RsB,U;QAAA,cA0RT,oBAAmB,cAAnB,EAAmB,sBAAnB,U;Q0L/sDIB,kB;Q1Lq7C2B,c0Lx7C3B,CAAC,Y  
AAS,CAAT,IAAc,qBAAf,KAA4C,Q1LktDG,M0LltDH,C1Lw7CjB,G0Lv7C3B,I1Lu7C2B,G0Lr7C3B,oBAxCmG  
,Q1LuvDpD,M0LvDoD,kBAwCnG,Y7LJD,yB6LIC,4B1L+sD+C,MA1RpB,U;UAA6C,+B;;M0L79ChF,OAOCK,  
S1Lo7CE,a0Lp7CF,EAAO,mBAAc,kBAAd,CAAP,EAA0C,IAA1C,CACA,W;K;IAjCI,8C;MAAA,qB;QAEg,IAA  
G,QAAH,EAAG,CAAH,C;UAEQ,IAAA,EAAG,OAAH,GAAY,cAAO,OAAAnB,C;YAHZ,OAGyC,c;;YAHZC,OAI  
oB,E;;UAJpB,OAoy,iBAAS,E;O;K;IAfjC,0C;MAKgC,sB;QAAA,SAAiB,M;MAC7C,OAYK,eAXA,OADL,uBA  
CK,EAAI,4BAAJ,CAWA,EAAa,IAAb,C;K;IAET,gC;MAAwC,uB;;Q9KmDtB,gC;QAAA,gC;QAAA,mB;QAAA,  
kB;QAAA,kB;QAAd,0D;UACI,I8KpD+C,CAAI,aAAH,U9KoDIC,iCAAK,KAAL,E8KpDkC,CAAG,C9KoDnD,C;  
YACI,sBAAO,K;YAAP,wB;;QAGR,sBAAO,E;;;Mf3CA,4B;M6Lb6B,OAA8C,OAMM,EAAY,GAAC,gBAAd,GA  
A0B,E;K;IAGpF,wC;MAAkB,W;K;IAC9B,oD;MAAA,uB;QAaKB,wBAAS,I;O;K;IAFvC,mC;MACI,IAAA,M7K  
kMgD,YAAU,C6KIM1D,C;QAD4C,OACxB,wB;;QADwB,OAEP,cK;K;mBAGZ,yB;M1L86CA,+D;MAoSA,wE  
;M0LltDA,sF;QAKI,gBAAGb,2B;Q1Lg7CT,kBAAoB,gB;QAoSd,gB;QADb,YAAY,C;QACC,2B;QAAb,OAAa,c  
AAb,C;UAAa,sB;UA1RsB,U;UAAA,cA0RT,oBAAmB,cAAnB,EAAmB,sBAAnB,U;U0L/sDIB,kB;U1Lq7C2B,c0  
Lx7C3B,CAAC,YAAS,CAAT,IAAc,qBAAf,KAA4C,Q1LktDG,I0LltDH,C1Lw7CjB,G0Lv7C3B,I1Lu7C2B,G0Lr  
7C3B,sC1L+sD+C,I0L/sD/C,a7LJD,yB6LIC,4B1L+sD+C,IA1RpB,U;YAA6C,6B;;Q0Lz7ChF,OAMK,S1Lo7CE,  
W0Lp7CF,EAAO,mBAAc,kBAAd,CAAP,EAA0C,IAA1C,CACA,W;O;KAbT,C;6EvEkSA,0B;MAGmE,OAAA,S  
AAK,gBAAO,GAAP,C;K;qFAExE,yB;MAAA,yD;MAAA,gC;QAO2B,gBAAhB,oB;QAAsB,atHrU7B,W;QsHqU  
A,OtHpUO,SsHoUqC,W;O;KAPhD,C;uFAUA,yB;MAAA,iE;MAAA,0C;QAQmC,gBAAXB,mBAAc,QAAd,C;QA  
A8B,atHhVrC,W;QsHgVA,OtH/UO,SsH+U6C,W;O;KARxD,C;IAWA,oC;MAIiB,Q;MAAb,wBAAa,KAAb,gB;Q  
AAa,WAAA,KAAb,M;QACI,yBAAO,IAAP,C;;MACJ,OAAO,S;K;IAGX,oC;MAIiB,Q;MAAb,wBAAa,KAAb,gB  
;QAAa,WAAA,KAAb,M;QACI,yBAAO,IAAP,C;;MACJ,OAAO,S;K;qFAGX,qB;MAG8D,gCAAO,EAAP,C;K;qF  
AE9D,4B;MAGkF,OAAA,yBAAO,KAAP,CALpB,gBAAO,EAAP,C;K;qFAO9D,4B;MAG4E,OAAA,yBAAO,KA  
AP,CAVd,gBAAO,EAAP,C;K;qFAY9D,4B;MAGyE,OAAA,yBAAO,KAAP,CAfX,gBAAO,EAAP,C;K;qFAiB9D  
,4B;MAG8E,OAAA,yBAAO,KAAP,CAPbB,gBAAO,EAAP,C;K;qFAsB9D,4B;MAGyE,OAAA,yBAAO,KAAP,  
CAzBX,gBAAO,EAAP,C;K;qFA2B9D,4B;MAG4E,OAAA,yBAAO,KAAP,CA9Bd,gBAAO,EAAP,C;K;I9H/a9D,  
iC;MAK0C,iCAAqB,EAArB,C;K;IAE1C,0C;MAQmB,Q;MAAA,qBAAL,SAAK,EAAY,KAAZ,C;MAAL,iB;QA  
A2B,OAAO,I;OAA5C,UAAU,I;MACV,IAAI,MAAM,sCAAK,UAAW,IAAwB,MAAM,sCAAK,UAAvC,C;QAAk  
D,OAAO,I;MACzD,OAAW,OAAJ,GAAL,C;K;IAGf,kC;MAK4C,kCAAsB,EAAtB,C;K;IAE5C,2C;MAQmB,Q;M  
AAA,qBAAL,SAAK,EAAY,KAAZ,C;MAAL,iB;QAA2B,OAAO,I;OAA5C,UAAU,I;MACV,IAAI,MAAM,uCAA  
M,UAAZ,IAAYB,MAAM,uCAAM,UAAzC,C;QAAoD,OAAO,I;MAC3D,OAAW,QAAJ,GAAL,C;K;IAGf,gC;MA  
KwC,gCAAOB,EAAPB,C;K;IAExC,yC;MAQI,WAAW,KAAX,C;MAEA,aAAa,SAAK,O;MACIB,IAAI,WAAU,C  
AAd,C;QAAiB,OAAO,I;MAExB,S;MACA,c;MACA,S;MAEA,gBAAGb,qBAAK,CAAL,C;MACHB,IAAI,YAAY,  
EAAbB,C;QACI,IAAI,WAAU,CAAd,C;UAAiB,OAAO,I;QAExB,QAAQ,C;QAER,IAAI,cAAa,EAAbB,C;UACI,a  
AAa,I;UACb,QAAQ,W;eACL,IAAI,cAAa,EAAbB,C;UACH,aAAa,K;UACb,QAAQ,W;;UAER,OAAO,I;;QAEX,Q  
AAQ,C;QACR,aAAa,K;QACb,QAAQ,W;;MAIZ,uBAAuB,S;MAEvB,qBAAQb,gB;MACrB,aAAa,C;MACb,aAA  
U,KAAV,MAAsB,MAAtB,M;QACI,YAAY,QAAQ,qBAAK,CAAL,CAAR,EAAbB,KAAjB,C;QAEZ,IAAI,QAAQ  
,CAAZ,C;UAAe,OAAO,I;QACtB,IAAI,SAAS,cAAb,C;UACI,IAAI,mBAAkB,gBAAtB,C;YACI,iBAAbB,QAAQ,  
KAAR,I;YAEjB,IAAI,SAAS,cAAb,C;cACI,OAAO,I;;YAGX,OAAO,I;;SAIf,6BAAU,KAAV,C;QAEA,IAAI,UAA  
S,QAAQ,KAAR,IAAT,CAAJ,C;UAA4B,OAAO,I;QAEnc,kBAAU,KAAV,I;;MAGJ,OAAW,UAAJ,GAAGB,MA  
AhB,GAA4B,CAAC,MAAD,I;K;IAGvC,iC;MAK0C,iCAAqB,EAArB,C;K;IAE1C,0C;MAQI,WAAW,KAAX,C;  
MAEA,aAAa,SAAK,O;MACIB,IAAI,WAAU,CAAd,C;QAAiB,OAAO,I;MAExB,S;MACA,c;MACA,S;MAEA,gB  
AAGb,qBAAK,CAAL,C;MACHB,IAAI,YAAY,EAAbB,C;QACI,IAAI,WAAU,CAAd,C;UAAiB,OAAO,I;QAExB,  
QAAQ,C;QAER,IAAI,cAAa,EAAbB,C;UACI,aAAa,I;UACb,gC;eACG,IAAI,cAAa,EAAbB,C;UACH,aAAa,K;UA  
Cb,6B;;UAEA,OAAO,I;;QAEX,QAAQ,C;QACR,aAAa,K;QACb,6B;;MAIJ,2C;MAEA,qBAAQb,gB;MACrB,e;M

ACA,aAAU,KAAV,MAAsB,MAAtB,M;QACI,YAA Y,QAAQ,qBAAK,CAAL,CAAR,EAAiB,KAAjB,C;QAEZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,IAAI,uBAAS,cAAT,KAAJ,C;UACI,IAAI,uBAaKB,gBAAlB,CAAJ,C;YACI,iBAAiB,8BAAQ,KAAR,E;YAEjB,IAAI,uBAAS,cAAT,KAAJ,C;cACI,OAAO,I;;YAGX,OAAO,I;;SAIf,6CAAU,KAAV,E;QAEA,IAAI,uBAAS,8BAAQ,KAAR,EAAT,KAAJ,C;UAA4B,OAAO,I;QAE nC,6CAAU,KAAV,E;;MAGJ,OAAW,UAAJ,GAAgB,MAAhB,GAA6B,MAAD,a;K;IAIvC,kC;MAAyD,MAAM,0BAAsB,6BAA0B,KAA1B,MAAtB,C;K;uEwBhI/D,yB;MAAA,oC;MAAA,uC;QAIi,iBAAiB,C;QACjB,eAAe,mBAAS,CAAT,I;QACf,iBAAiB,K;QAEjB,OAAO,cAAc,QAArB,C;UACI,YAAgB,CAAC,UAAAL,GAAiB,UAAjB,GAAiC,Q;UAC7C,YAAY,UAAU,iCAAK,KAAL,EA AV,C;UAEZ,IAAI,CAAC,UAAAL,C;YACI,IAAI,CAAC,KAAL,C;cACI,aAAa,I;;cAEb,0BAAc,CAAd,I;;YAEJ,IAAI,CAAC,KAAL,C;cACI,K;;cAEA,sBAAY,CAAZ,I;;;QAIZ,OAAO,8BAAY,UAAZ,EAAwB,WAAW,CAAX,IAAxB,C;O;KAZBX,C;yEA4BA,yB;MAAA,8B;MA5BA,oC;MA4BA,uC;QAIK,Q;QAA sB,kBAAtB,2D;QA5BD,iBAAiB,C;QACjB,eAAe,qBAAS,CAAT,I;QACf,iBAAiB,K;QAEjB,OAAO,cAAc,QAArB,C;UACI,YAAgB,CAAC,UAAAL,GAAiB,UAAjB,GAAiC,Q;UAC7C,YAsBwB,SAtBZ,CAAU,mCAAK,KAAL,EA AV,C;UAEZ,IAAI,CAAC,UAAAL,C;YACI,IAAI,CAAC,KAAL,C;cACI,aAAa,I;;cAEb,0BAAc,CAAd,I;;YAEJ,IAAI,CAAC,KAAL,C;cACI,K;;cAEA,sBAAY,CAAZ,I;;;QAWZ,OAPO,gCAAY,UAAZ,EAAwB,WAAW,CAAX,IAAxB,CAOGC,W;O;KAJ3C,C;iFAMA,yB;MAAA,mD;MAAA,oC;MAAA,uC;QAIuB,UAAAL,MAAK,EAAL,MAAK,EAAL,M;QAAK,mBAAL,SAAK,C;QAAL,mB;QAAA,kB;QAAA,kB;QAAd,0D;UACI,IAAI,CAAC,UAAU,iCAAK,KAAL,EA AV,CAAL,C;YACI,OAAO,8BAAY,KAAZ,EAAMb,gBAAnB,C;QAEf,OAAO,E;O;KARX,C;mFAWA,yB;MAAA,8B;MAXA,mD;MAAA,oC;MAWA,uC;QAIK,Q;QAAsB,kBAAtB,2D;QAAsB,oB;;UAXJ,kC;UAAA,qBAAL,WAAK,C;UAAAL,qB;UAAA,oB;UAAA,oB;UAAAd,0D;YACI,IAAI,CAUyB,SAVxB,CAAU,mCAAK,KAAL,EA AV,CAAL,C;cACI,mBAAO,gCAAY,KAAZ,EAAMb,kBAAnB,C;cAAP,qB;aAER,mBAAO,E;;;QAOP,OAA4C,2B;O;KAJhD,C;6EAMA,yB;MAAA,mD;MAAA,+C;MAAA,oC;MAAA,uC;QAIkB,Q;QAAA,OAAa,SAAR,YAAL,SAAK,CAAQ,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,IAAI,CAAC,UAAU,iCAAK,KAAL,EA AV,CAAL,C;YACI,OAAO,8BAAY,CAAZ,EAAe,QAAQ,CAAR,IAAf,C;;QAEf,OAAO,E;O;KARX,C;+EAWA,yB;MAAA,8B;MAXA,mD;MAAA,+C;MAAA,oC;MAWA,uC;QAIK,Q;QAAsB,kBAAtB,2D;QAAsB,kB;;UAXT,U;UAAA,SAAa,SAAR,YAAL,WAAK,CAAQ,CAAb,W;UAAAd,OAAc,gBAAd,C;YAAc,yB;YACV,IAAI,CAUuB,SAVtB,CAAU,mCAAK,KAAL,EA AV,CAAL,C;cACI,iBAAO,gCAAY,CAAZ,EAAe,QAAQ,CAAR,IAAf,C;cAAP,mB;;UAER,iBAAO,E;;;QAOP,OAA0C,yB;O;KAJ9C,C;IAMA,kC;MAhEI,iBAAiB,C;MACjB,eAAe,mBAAS,CAAT,I;MACf,iBAAiB,K;MAEjB,OAAO,cAAc,QAArB,C;QACI,YAAgB,CAAC,UAAAL,GAAiB,UAAjB,GAAiC,Q;QAC7C,YA6DgE,4BA7D1C,iCAAK,KAAL,EA6D0C,E;QA3DhE,IAAI,CAAC,UAAAL,C;UACI,IAAI,CAAC,KAAL,C;YACI,aAAa,I;;YAEb,0BAAc,CAAd,I;;UAEJ,IAAI,CAAC,KAAL,C;YACI,K;;YAEA,sBAAY,CAAZ,I;;;MAkDiD,OA9CtD,8BAAY,UAAZ,EAAwB,WAAW,CAAX,IAAxB,C;K;IAGDX,kC;MAzCK,Q;MAAsB,kBAAtB,2D;MA5BD,iBAAiB,C;MACjB,eAAe,qBAAS,CAAT,I;MACf,iBAAiB,K;MAEjB,OAAO,cAAc,QAArB,C;QACI,YAAgB,CAAC,UAAAL,GAAiB,UAAjB,GAAiC,Q;QAC7C,YAkEoD,4BAIE9B,mCAAK,KAAL,EAkE8B,E;QAhEpD,IAAI,CAAC,UAAAL,C;UACI,IAAI,CAAC,KAAL,C;YACI,aAAa,I;;YAEb,0BAAc,CAAd,I;;UAEJ,IAAI,CAAC,KAAL,C;YACI,K;;YAEA,sBAAY,CAAZ,I;;;MAuDqC,OAnD1C,gCAAY,UAAZ,EAAwB,WAAW,CAAX,IAAxB,CAOGC,W;K;IA8C3C,uC;MAGsE,oB;;QA3C/C,gC;QAAA,gC;QAAL,mB;QAAA,kB;QAAA,kB;QAAd,0D;UACI,IAAI,CA0CsE,4BA1C3D,iCAAK,KAAL,EA0C2D,EA1C1E,C;YACI,mBAAO,8BAAY,KAAZ,EAAMb,gBAAnB,C;YAAP,qB;;QAER,mBAAO,E;;;MAuC2D,uB;K;IAEtE,uC;MAICK,Q;MAAsB,kBAAtB,2D;MAAsB,oB;;QAXJ,kC;QAAA,wBAAL,WAAK,C;QAAL,qB;QAAA,oB;QAAA,oB;QAAd,0D;UACI,IAAI,CA+C0D,4BA/C/C,mCAAK,KAAL,EA+C+C,EA/C9D,C;YACI,mBAAO,gCAAY,KAAZ,EAAMb,kBAAnB,C;YAAP,qB;;QAER,mBAAO,E;;;MA4C+C,OArCV,2B;K;IAuChD,qC;MAGoE,kB;;QApCID,Q;QAAA,OAAa,WAAR,yBAAQ,CAAb,W;QAAd,OAAc,cAAAd,C;UAAc,uB;UACV,IAAI,CAMcKE,4BAnCvD,iCAAK,KAAL,EA mCuD,EAnCtE,C;YACI,iBAAO,8BAAY,CAAZ,EAAe,QAAQ,CAAR,IAAf,C;YAAP,mB;;QAER,iBAAO,E;;;MAGCyD,qB;K;IAEpE,qC;MA3BK,Q;MAAsB,kBAAtB,2D;MAAsB,kB;;QAXT,U;QAAA,SAAa,WAAR,eAAL,WAAK,CAAQ,CAAb,W;QAAd,OAAc,gBAAd,C;UAAc,yB;UACV,IAAI,CAwCsD,4BAx3C,mCAAK,KAAL,EAwC2C,EAxC1D,C;YACI,iBAAO,gCAAY,CAAZ,EAAe,QAAQ,CAAR,IAAf,C;YAAP,mB;;QAER,iBAAO,E;;;MAqC6C,OA9BV,yB;K;IAGC9C,2B;MA9FI,iBAAiB,C;MACjB,eAAe,mBAAS,CAAT,I;MACf,iBAAiB,K;MAEjB,OAAO,cAAc,QAArB,C;QACI,YAAgB,CAAC,UAAAL,GAAiB,UAAjB,GAAiC,Q;QAC7C,mCAAsB,iCAAK,KAAL,EAAtB,E;QAEA,IAA

I,CAAC,UAAL,C;UACI,IAAI,CAAC,KAAL,C;YACI,aAAa,I;;YAEb,0BAAc,CAAd,I;;UAEJ,IAAI,CAAC,KAAL,C;YACI,K;;YAEA,sBAAY,CAAZ,I;;;MAGf+B,OA5EpC,8BAAY,UA AZ,EAawB,WAaw,CAAX,IAAxB,C;K;yEA8EX,yB;MAAA,8B;MAAA,qC;MAAA,4B;QAI2C,Q;QAAD,OAAuB,KAAtB,2DAAsB,CAAO,W;O;KAJxE,C;IAMA,gC;MAGoD,oB;;QA1E7B,gC;QAAA,gC;QAAL,mB;QAAA,kB;QAAA,kB;QAAd,0D;UACI,IAAI,wBAAW,iCAAK,KAAL,EAAX,EA AJ,C;YACI,mBAAO,8BAAY,KAAZ,EAAMb,gBAAnB,C;YAAP,qB;;QAER,mBAAO,E;;;MAsEyC,uB;K;mFAEpD,yB;MAAA,8B;MAAA,+C;MAAA,4B;QAIgD,Q;QAAD,OAAuB,UAAtB,2DAAsB,CAAY,W;O;KAJIF,C;IAMA,8B;MAGkD,kB;;QApEhC,Q;QAAA,OAAa,WAAR,yBAAQ,CAAb,W;QAAd,OA Ac,cAAAd,C;UAAc,uB;UACV,IAAI,wBAAW,iCAAK,KAAL,EAAX,EA AJ,C;YACI,iBAAO,8BAAY,CAAZ,EA Ae,QAAQ,CAAR,IAAf,C;YAAP,mB;;QAER,iBAAO,E;;;MAGEuC,qB;K;+EAEID,yB;MAAA,8B;MAAA,2C;MAA A,4B;QAI8C,Q;QAAD,OAAuB,QAAtB,2DAAsB,CAAU,W;O;KAJ9E,C;IAMA,8C;MAU8C,uB;QAAA,UAAgB, E;MAO5C,Q;MANd,IAAI,SAAS,CAAb,C;QACI,MAAM,gCAAYB,oBAAiB,MAAjB,wBAAzB,C;MACV,IAAI,U AAU,SAAK,OAAAnB,C;QACI,OAAy,mBAAL,SAAK,EAAY,CAAZ,EA Ae,SAAK,OAApB,C;MAEhB,SAAS,mB AAc,MAAd,C;MACK,gBAAS,SAAK,OAAAd,I;MAAd,aAAU,CAAV,iB;QACI,EAAG,gBAAO,OAAp,C;MACP,E AAG,gBAAO,SAAP,C;MACH,OAAO,E;K;IAGX,gD;MASwC,uB;QAAA,UAAgB,E;MACnD,Q;MAAD,OAAuB, SAAtB,6DAAsB,EAAS,MAAT,EA AiB,OAAjB,CAA0B,W;K;IAErD,4C;MAU4C,uB;QAAA,UAAgB,E;MAQ1C, Q;MAPd,IAAI,SAAS,CAAb,C;QACI,MAAM,gCAAYB,oBAAiB,MAAjB,wBAAzB,C;MACV,IAAI,UAAU,SA A K,OAAAnB,C;QACI,OAAy,mBAAL,SAAK,EAAY,CAAZ,EA Ae,SAAK,OAApB,C;MAEhB,SAAS,mB AAc,MAAd,C;MACT,EAAG,gBAAO,SAAP,C;MACW,gBAAS,SAAK,OAAAd,I;MAAd,aAAU,CAAV,iB;QACI,EAAG,gB AAO,OAAp,C;MACP,OAAO,E;K;IAGX,8C;MASsC,uB;QAAA,UAAgB,E;MACjD,Q;MAAD,OAAuB,OAAtB,6D AAsB,EA AO,MAAP,EA Ae,OA Af,CAAwB,W;K;2FAEnD,qB;MAWI,OAAO,qBAAgB,SAAK,OAAL,KAAe,C;K ;+EAG1C,qB;MAMoD,4BAAU,C;K;sFAE9D,qB;MAMuD,0BAAS,C;K;mFAMhE,yB;MAAA,2C;MAAA,4B;QA MuD,QAAC,kB;O;KANxD,C;yFAQA,yB;MAAA,2C;MAAA,4B;QAWI,OAAO,qBAAqB,QAAL,SAAK,C;O;KA XhC,C;IAiB4D,+C;MAAA,kC;MAAS,uB;MACjE,eAAoB,C;K;gDAEpB,Y;MAA2C,gB;MAAA,iE;MAAJ,4C;K;+ CAEvC,Y;MAAYC,sBAAQ,yB;K;;IARrD,+B;MAG4D,4C;K;+EAQ5D,qB;MAE8C,uCAAQ,E;K;+EAETd,mC;M ASI,OA5DgD,qBAAU,CA4D1D,GAAe,cAAf,GAAMC,S;K;6EAEvC,yB;MAAA,2C;MAAA,0C;QASI,OAAI,kBA AJ,GAAe,cAAf,GAAMC,S;O;KATvC,C;IAeI,mC;MAAQ,uBAAG,mBAAS,CAAT,IAAH,C;K;IAMR,qC;MAAQ, OAAA,SAAK,OAAL,GAAc,CAAd,I;K;IAEZ,8C;MAIuB,Q;MAAA,0BAAS,CAAT,I;MAAnB,OAAgB,CAAT,8B ACgB,gBAAZ,qBAAK,KAAL,CAAY,CADhB,IAEoB,eAAhB,qBAAK,QAAQ,CAAR,IAAL,CAAgB,C;K;IAG/B, uC;MAGuD,ONpKyC,oBMoK/B,KAAM,MNpKyB,EMoKIB,KAAM,aAAN,GAAqB,CAArB,INpKkB,C;K;IMsK hG,yC;MAGqE,qCAAY,KAAM,MAAIB,EAAYB,KAAM,aAAN,GAAqB,CAArB,IAAzB,C;K;uFAErE,iC;MAS2E ,2BAAY,KAAZ,EAAMb,GAAAnB,C;K;mFAE3E,2C;MAOOD,wB;QAAA,WAAgB,gB;MAAkB,OAAA,8BAAY,U AAZ,EAawB,QAAXB,CAAKC,W;K;IAE9H,uC;MAG6D,OAAA,8BAAY,KAAM,MAAIB,EAAYB,KAAM,aAAN ,GAAqB,CAArB,IAAzB,CAAiD,W;K;IAE9G,sE;MAImD,qC;QAAA,wBAAGC,S;MAC/E,YAAY,sBAAQ,SAAR, C;MACZ,OAAW,UAAS,EAAPB,GAAwB,qBAAxB,GN1M4F,oBM0M/B,CN1M+B,EM0M5B,KN1M4B,C;K;IM 6MhG,wE;MAIqD,qC;QAAA,wBAAGC,S;MACjF,YAAY,sBAAQ,SAAR,C;MACZ,OAAW,UAAS,EAAPB,GAA wB,qBAAxB,GNnN4F,oBMmN/B,CNnN+B,EMmN5B,KNnN4B,C;K;IMsNhG,qE;MAIkD,qC;QAAA,wBAAGC, S;MAC9E,YAAY,sBAAQ,SAAR,C;MACZ,OAAW,UAAS,EAAPB,GAAwB,qBAAxB,GN5N4F,oBM4N/B,QAA Q,CAAR,IN5N+B,EM4NpB,gBN5NoB,C;K;IM+NhG,uE;MAIoD,qC;QAAA,wBAAGC,S;MACHF,YAAY,sBAAQ ,SAAR,C;MACZ,OAAW,UAAS,EAAPB,GAAwB,qBAAxB,GNrO4F,oBMqO/B,QAAQ,SAAU,OAAIB,INrO+B, EMqOL,gBNrOK,C;K;IMwOhG,0E;MAIuD,qC;QAAA,wBAAGC,S;MACnF,YAAY,0BAAY,SAAZ,C;MACZ,OA AW,UAAS,EAAPB,GAAwB,qBAAxB,GN9O4F,oBM8O/B,CN9O+B,EM8O5B,KN9O4B,C;K;IMiPhG,4E;MAIy D,qC;QAAA,wBAAGC,S;MACrF,YAAY,0BAAY,SAAZ,C;MACZ,OAAW,UAAS,EAAPB,GAAwB,qBAAxB,G NvP4F,oBMuP/B,CNvP+B,EMuP5B,KNvP4B,C;K;IM0PhG,yE;MAIsD,qC;QAAA,wBAAGC,S;MACIF,YAAY,0 BAAY,SAAZ,C;MACZ,OAAW,UAAS,EAAPB,GAAwB,qBAAxB,GNhQ4F,oBMgQ/B,QAAQ,CAAR,INhQ+B,E MgQpB,gBNhQoB,C;K;IMmQhG,2E;MAIwD,qC;QAAA,wBAAGC,S;MACpF,YAAY,0BAAY,SAAZ,C;MACZ, OAAW,UAAS,EAAPB,GAAwB,qBAAxB,GNzQ4F,oBMyQ/B,QAAQ,SAAU,OAAIB,INzQ+B,EMyQL,gBNzQK ,C;K;IM4QhG,oE;MAOI,IAAI,WAaw,UAaf,C;QACI,MAAM,8BAA0B,gBAAa,QAAb,oCAAKD,UAAID,OAAI B,C;MACV,SAAS,sB;MACT,EAAG,qBAAy,SAAZ,EAakB,CAAlB,EAaqB,UAArB,C;MACH,EAAG,gBAAO,

WAAP,C;MACH,EAAG,qBAAY,SAAZ,EAakB,QAAIB,EAA4B,gBAA5B,C;MACH,OAAO,E;K;yFAGX,yB;M  
AAA,8B;MAAA,qD;MAAA,+D;QAOK,Q;QAAD,OAAuB,aAAtB,2DAAsB,EAAa,UAAb,EAAYB,QAazB,EAA  
mC,WAAnc,CAAGD,W;O;KAP3E,C;IASA,uD;MAOI,+BAAa,KAAM,MAAnB,EAA0B,KAAM,aAN,GAAqB,  
CAArB,IAA1B,EAakD,WAAID,C;K;yFAEJ,yB;MAAA,8B;MAAA,qD;MAAA,gD;QAOK,Q;QAAD,OAAuB,aA  
AtB,2DAAsB,EAAa,KAAb,EAAoB,WAApB,CAAIc,W;O;KAP5D,C;IASA,sD;MASI,IAAI,WAAW,UAAf,C;QA  
CI,MAAM,8BAA0B,gBAAa,QAAb,oCAakD,UAAID,OAA1B,C;MAEV,IAAI,aAAY,UAAhB,C;QACI,OAAy,m  
BAAL,SAAK,EAAY,CAAZ,EAAe,gBAAf,C;MAEHb,SAAS,mBAAc,oBAAU,QAAV,GAAqB,UAArB,KAAc,C;  
MACT,EAAG,qBAAY,SAAZ,EAakB,CAAIb,EAAqB,UAArB,C;MACH,EAAG,qBAAY,SAAZ,EAakB,QAAIB,  
EAA4B,gBAA5B,C;MACH,OAAO,E;K;uFAGX,yB;MAAA,8B;MAAA,mD;MAAA,kD;QASK,Q;QAAD,OAAuB  
,YAAtB,2DAAsB,EAAY,UAAZ,EAawB,QAAxB,CAAKC,W;O;KAT7D,C;IAWA,yC;MAKqE,8BAAY,KAAM,  
MAAIB,EAAYB,KAAM,aAN,GAAqB,CAArB,IAAZB,C;K;uFAErE,yB;MAAA,8B;MAAA,mD;MAAA,mC;QA  
OK,Q;QAAD,OAAuB,YAAtB,2DAAsB,EAAY,KAAZ,CAAmB,W;O;KAP9C,C;IASA,yC;MAKI,IAAI,wBAAW,  
MAAX,CAAJ,C;QACI,OAAO,8BAAY,MAAO,OAAAnB,EAA2B,gBAA3B,C;OAEX,OAAO,8BAAY,CAAZ,EAA  
e,gBAAf,C;K;IAGX,2C;MAKI,IAAI,wBAAW,MAAX,CAAJ,C;QACI,ON3XyE,oBM2XxD,MAAO,ON3XiD,C;O  
M6X7E,OAAO,S;K;IAGX,yC;MAKI,IAAI,sBAAS,MAAT,CAAJ,C;QACI,OAAO,8BAAY,CAAZ,EAAe,mBAAS  
,MAAO,OAAhB,IAAf,C;OAEX,OAAO,8BAAY,CAAZ,EAAe,gBAAf,C;K;IAGX,2C;MAKI,IAAI,sBAAS,MAAT  
,CAAJ,C;QACI,ON9YwF,oBM8YvE,CN9YuE,EM8YpE,mBAAS,MAAO,OAAhB,IN9YoE,C;OMgZ5F,OAAO,S;  
K;IAGX,sD;MAMI,IAAK,qBAAU,MAAO,OAAP,GAAgB,MAAO,OAAvB,IAAV,CAAD,IAA6C,wBAAW,MAA  
X,CAA7C,IAAmE,sBAAS,MAAT,CAAvE,C;QACI,OAAO,8BAAY,MAAO,OAAAnB,EAA2B,mBAAS,MAAO,O  
AAhB,IAA3B,C;OAEX,OAAO,8BAAY,CAAZ,EAAe,gBAAf,C;K;IAGX,wD;MAMI,IAAK,qBAAU,MAAO,OA  
AP,GAAgB,MAAO,OAAvB,IAAV,CAAD,IAA6C,wBAAW,MAAX,CAA7C,IAAmE,sBAAS,MAAT,CAAvE,C;  
QACI,ONtawF,oBMsavE,MAAO,ONtagE,EMsaxD,mBAAS,MAAO,OAAhB,INtawD,C;OMwa5F,OAAO,S;K;IA  
GX,mD;MAKmf,oCAakB,SAAIb,EAA6B,SAA7B,C;K;IAEnF,mD;MAKuE,sCAakB,SAAIb,EAA6B,SAA7B,C  
;K;IAEvE,iF;MAIsE,qC;QAAA,wBAAGC,S;MACIG,YAAY,sBAAQ,SAAR,C;MACL,Q;MAAA,IAAI,UAAS,EA  
Ab,C;QAAA,OAAiB,qB;;QA5JvB,U;QA4JM,OA5JgB,aAAtB,+DAAsB,EA4JyC,CA5JzC,EA4J4C,KA5J5C,EA4J  
mD,WA5JnD,CAAGD,W;;MA4JvE,W;K;IAGJ,mF;MAIwE,qC;QAAA,wBAAGC,S;MACpG,YAAY,sBAAQ,SAA  
R,C;MACL,Q;MAAA,IAAI,UAAS,EAAb,C;QAAA,OAAiB,qB;;QArKvB,U;QAqKM,OArKgB,aAAtB,+DAAsB,  
EAqKyC,CARkZC,EAqK4C,KArK5C,EAqKmD,WArKnD,CAAGD,W;;MAqKvE,W;K;IAGJ,gF;MAIqE,qC;QAA  
A,wBAAGC,S;MACjG,YAAY,sBAAQ,SAAR,C;MACL,Q;MAAA,IAAI,UAAS,EAAb,C;QAAA,OAAiB,qB;;QA  
A2B,iBAAa,QAAQ,CAAR,I;QAAb,eAAwB,gB;QA9K1E,U;QA8KM,OA9KgB,aAAtB,+DAAsB,EAAa,UAAb,E  
AAyB,QAazB,EA8K4D,WA9K5D,CAAGD,W;;MA8KvE,W;K;IAGJ,kF;MAIuE,qC;QAAA,wBAAGC,S;MACnG,  
YAAY,sBAAQ,SAAR,C;MACL,Q;MAAA,IAAI,UAAS,EAAb,C;QAAA,OAAiB,qB;;QAA2B,iBAAa,QAAQ,SA  
AU,OAAIB,I;QAAb,eAAuC,gB;QAvLzF,U;QAUlM,OAvlGb,aAAtB,+DAAsB,EAAa,UAAb,EAAYB,QAazB,E  
AuL2E,WAvL3E,CAAGD,W;;MAuLvE,W;K;IAGJ,oF;MAI2E,qC;QAAA,wBAAGC,S;MACvG,YAAY,0BAAY,S  
AAZ,C;MACL,Q;MAAA,IAAI,UAAS,EAAb,C;QAAA,OAAiB,qB;;QAA2B,iBAAa,QAAQ,SAAU,OAAIB,I;QA  
Ab,eAAuC,gB;QAhMzF,U;QAgMM,OAhmGb,aAAtB,+DAAsB,EAAa,UAAb,EAAYB,QAazB,EAyM2E,WAhM  
3E,CAAGD,W;;MAgMvE,W;K;IAGJ,sF;MAIyE,qC;QAAA,wBAAGC,S;MACrG,YAAY,0BAAY,SAAZ,C;MACL  
,Q;MAAA,IAAI,UAAS,EAAb,C;QAAA,OAAiB,qB;;QAA2B,iBAAa,QAAQ,CAAR,I;QAAb,eAAwB,gB;QAZM1  
E,U;QAYMM,OAzmGb,aAAtB,+DAAsB,EAAa,UAAb,EAAYB,QAazB,EAyM4D,WAZM5D,CAAGD,W;;MAyM  
vE,W;K;IAGJ,qF;MAI0E,qC;QAAA,wBAAGC,S;MACtG,YAAY,0BAAY,SAAZ,C;MACL,Q;MAAA,IAAI,UAA  
S,EAAb,C;QAAA,OAAiB,qB;;QAINvB,U;QAKNM,OAInGb,aAAtB,+DAAsB,EAKNyC,CAINzC,EAKN4C,KAIN  
5C,EAKNmD,WAINnD,CAAGD,W;;MAkNvE,W;K;IAGJ,uF;MAI4E,qC;QAAA,wBAAGC,S;MACxG,YAAY,0B  
AAY,SAAZ,C;MACL,Q;MAAA,IAAI,UAAS,EAAb,C;QAAA,OAAiB,qB;;QA3NvB,U;QA2NM,OA3NgB,aAAtB  
,+DAAsB,EA2NyC,CA3NzC,EA2N4C,KA3N5C,EA2NmD,WA3NnD,CAAGD,W;;MA2NvE,W;K;+EAOJ,yC;MA  
QoF,OAAA,KAAM,iBAAQ,SAAR,EAAC,WAAc,C;K;+EAE1F,uC;MAOI,OAAA,KAAM,iBAAQ,SAAR,EAAC,S  
AAd,C;K;yFAEV,yC;MAMyF,OAAA,KAAM,sBAAa,SAAb,EAAMb,WAAAnB,C;K;+FAE/F,yB;MAAA,oC;MAA  
A,gC;MAAA,uC;QAEW,Q;QAAA,IApe4C,mBAAS,CAoerD,C;uBAakB,oBAAU,iCAAK,CAAL,EAavE,UAA  
A,YNljBoD,oBMkjBrB,CNljBqB,C;UMkjBtE,OLrjBwD,2BAAL,GAakB,K;;UKqjBrE,OAAyD,S;QAAhE,W;O;K

AfJ,C;iGakBA,yB;MAAA,oC;MAAA,uC;QAeI,OatfmD,mBAAS,CAsf5D,GAAyB,UAAU,iCAAK,CAAL,EAA  
V,CAAmB,WAAAnB,GNpkBoD,oBMokBV,CNpkBU,CMokB7E,GAA2E,S;O;Kaf/E,C;+EAmBA,4B;MAIsE,OA  
AA,KAAM,iBAAQ,SAAR,C;K;IAE5E,0F;MAKI,IAAK,cAAc,CAAf,IAAsB,aAAa,CAAnC,IAA0C,cAAa,SAAK,  
OoAL,GAAc,MAAd,IAAb,CAA1C,IAAiF,eAAc,KAAM,OAAN,GAAe,MAAf,IAAd,CAArF,C;QACI,OAAO,K;  
OAGX,iBAAc,CAAd,UAAsB,MAAtB,U;QACI,IAAI,CAA0B,SAAzB,qBAAK,aAAa,KAAb,IAAL,CAAYB,EAA  
O,iBAAM,cAAc,KAAAd,IAAN,CAAP,EAAmC,UAAAnC,CAA9B,C;UACI,OAAO,K;;MAEf,OAAO,I;K;IAGX,mD;  
MAG+C,0B;QAAA,aAAsB,K;MACjE,OAAA,SAAK,OAAL,GAAc,CAAd,IAA2B,SAAR,qBAAK,CAAL,CAAQ,  
EAAO,IAAP,EAAa,UAAb,C;K;IAE/B,iD;MAG6C,0B;QAAA,aAAsB,K;MAC/D,OAAA,SAAK,OAAL,GAAc,C  
AAAd,IAAmC,SAAhB,qBAAK,2BAAL,CAAgB,EAAO,IAAP,EAAa,UAAb,C;K;IAEvC,qD;MAGyD,0B;QAAA,a  
AAsB,K;MAC3E,IAAI,CAAC,UAAD,IAAe,6BAAf,IAAiC,0BAArC,C;QACI,OAAy,WAAAL,SAAK,EAAW,MA  
AX,C;;QAEZ,OAAO,6BAAkB,CAAIB,EAAqB,MAArB,EAA6B,CAA7B,EAAGC,MAAO,OAAvC,EAA+C,UAA/  
C,C;K;IAGf,iE;MAG0E,0B;QAAA,aAAsB,K;MAC5F,IAAI,CAAC,UAAD,IAAe,6BAAf,IAAiC,0BAArC,C;QAC  
I,OAAy,aAAL,SAAK,EAAW,MAAX,EAAmB,UAAAnB,C;;QAEZ,OAAO,6BAAkB,UAAIB,EAA8B,MAA9B,EA  
AsC,CAAtC,EAAyC,MAAO,OAAd,EAAwD,UAAxD,C;K;IAGf,mD;MAGuD,0B;QAAA,aAAsB,K;MACzE,IA  
AI,CAAC,UAAD,IAAe,6BAAf,IAAiC,0BAArC,C;QACI,OAAy,SAAL,SAAK,EAAS,MAAT,C;;QAEZ,OAAO,6  
BAAkB,mBAAS,MAAO,OAAd,IAAIB,EAA0C,MAA1C,EAAd,CAAID,EAAqD,MAAO,OAA5D,EAAoE,UA  
ApE,C;K;IAMf,wD;MAQ8D,0B;QAAA,aAAsB,K;MACHf,qBfjnBO,MAAO,KeinBa,SAAK,OfjnBIB,EinB0B,K  
AAM,OfjnBhC,C;MemnBd,QAAQ,C;MACR,OAAO,IAAI,cAAJ,IAA8B,SAAR,qBAAK,CAAL,CAAQ,EAAO,iB  
AAM,CAAN,CAAP,EAA8B,UAA9B,CAArC,C;QACI,a;;MAEJ,IAAS,mBAAL,SAAK,EAAmB,IAAI,CAAJ,IAA  
nB,CAAL,IAAwC,mBAAN,KAAM,EAAmB,IAAI,CAAJ,IAAnB,CAA5C,C;QACI,a;OAEJ,OAAO,8BAAY,CAA  
Z,EAAe,CAAf,CAAkB,W;K;IAG7B,wD;MAQ8D,0B;QAAA,aAAsB,K;MACHf,iBAAiB,SAAK,O;MACtB,kBAA  
kB,KAAM,O;MACxB,qBfxoBO,MAAO,KewoBa,UfxoBb,EewoByB,WfxoBzB,C;Me0oBd,QAAQ,C;MACR,OA  
AO,IAAI,cAAJ,IAA+C,SAAzB,qBAAK,aAAa,CAAb,GAAiB,CAAjB,IAAL,CAAYB,EAAO,iBAAM,cAAc,CAA  
d,GAAkB,CAAIB,IAAN,CAAP,EAAGD,UAAhD,CAAtD,C;QACI,a;;MAEJ,IAAS,mBAAL,SAAK,EAAmB,aAAa  
,CAAb,GAAiB,CAAjB,IAAnB,CAAL,IAAqD,mBAAN,KAAM,EAAmB,cAAc,CAAd,GAAkB,CAAIB,IAAnB,C  
AAzD,C;QACI,a;OAEJ,OAAO,8BAAY,aAAa,CAAb,IAAZ,EAA4B,UAA5B,CAAwC,W;K;IAMnD,8D;MAQqD,  
0B;QAAA,aAakB,C;MAAG,0B;QAAA,aAAsB,K;MAMnE,UAAkB,M;MAL3C,IAAI,CAAC,UAAD,IAAe,KAA  
M,OAAN,KAAC,CAA7B,IAAKC,6BAAtC,C;QACI,WAAiB,SAAN,KAAM,C;QACjB,ONjtBwF,kB6G3ME,oBvG  
45BrE,IuG55BqE,C7G2MF,EMitB7D,UNjtB6D,C;OMotBnE,uBAAX,UAAW,EAAC,CAAd,C;MAAkB,oC;kBAA  
3C,gD;QACI,kBAAkB,qBAAI,KAAJ,C;QACR,c;;UjCikXE,U;UAAhB,4BiCjkXQ,KjCikXR,kB;YAAgB,cAAhB,  
UiCjkXQ,KjCikXR,S;YAAAsB,IiCjkXC,SAAH,UjCikXgB,oBiCjkXhB,CAAG,0BjCikXD,C;cAAwB,aAAO,I;cAA  
P,e;;UAC9C,aAAO,K;;QiCikXH,e;UACI,OAAO,K;;MAEf,OAAO,E;K;IAGX,KE;MASyD,0B;QAAA,aAakB,2B;  
MAAW,0B;QAAA,aAAsB,K;MACxG,IAAI,CAAC,UAAD,IAAe,KAAM,OAAN,KAAC,CAA7B,IAAKC,6BAAtC  
,C;QACI,WAAiB,SAAN,KAAM,C;QACjB,ONruB4F,sB6G3MM,oBvGg7BzE,IuGh7ByE,C7G2MN,EMquB7D,U  
NruB6D,C;mBMyuBhG,iBAAYB,eAAX,UAAW,EAAa,2BAAb,CAAzB,WAAwD,CAAxD,U;QACI,kBAAkB,qB  
AAI,KAAJ,C;QACR,c;;UjCyIXE,Q;UAAhB,wBiCziXQ,KjCyiXR,gB;YAAgB,cAAhB,UiCziXQ,KjCyiXR,O;YA  
AsB,IiCziXC,SAAH,UjCyiXgB,oBiCziXhB,CAAG,0BjCyiXD,C;cAAwB,aAAO,I;cAAP,e;;UAC9C,aAAO,K;;Qi  
C1iXH,e;UACI,OAAO,K;;MAGf,OAAO,E;K;IAIX,8E;MAA2G,oB;QAAA,OAAgB,K;MAOrG,UAKA,M;MAXI  
B,cAAkB,CAAC,IAAL,GACV,aAAW,gBAAX,UAAW,EAAC,CAAd,CAAX,EAAS,cAAT,QAAS,EAAa,gBAAb  
,CAAtC,CADU,GAGV,SAAW,eAAX,UAAW,EAAa,2BAAb,CAAX,EAAmD,gBAAT,QAAS,EAAc,CAAd,CAA  
nD,C;MAEJ,IAAI,iCAAkB,yBAAtB,C;QACkB,yB;QAAd,OAAc,cAAd,C;UAAc,uB;UACV,IAAU,cAAN,KAAM  
,EAAC,CAAd,EAAiB,SAAjB,EAAuB,KAAvB,EAA8B,KAAM,OAAP,C,EAA4C,UAA5C,CAAV,C;YACI,OAAO,  
K;;QAGD,2B;QAAd,OAAc,gBAAd,C;UAAc,2B;UACV,IAAU,kBAAN,KAAM,EAAkB,CAAIB,EAAqB,SAArB,  
EAA2B,OAA3B,EAAkC,KAAM,OAAxC,EAAGD,UAAhD,CAAV,C;YACI,OAAO,O;;MAGnB,OAAO,E;K;IAG  
X,qE;MAUsB,UAMA,M;MAfIB,IAAI,CAAC,UAAD,IAAe,OAAQ,KAAR,KAAgB,CAAnC,C;QACI,aAAqB,UA  
AR,OAAQ,C;QACrB,YAAgB,CAAC,IAAL,GAAW,sBAAQ,MAAR,EAAgB,UAAhB,CAAX,GAA4C,0BAAY,M  
AAZ,EAAoB,UAApB,C;QACxD,OAAW,QAAQ,CAAZ,GAAe,IAAf,GAAyB,UAAAS,MAAT,C;OAGpC,cAAkB,  
CAAC,IAAL,GAAW,aAAW,gBAAX,UAAW,EAAC,CAAd,CAAX,EAA6B,gBAA7B,CAAX,GAAoD,SAAW,eA

AX,UAAW,EAAa,2BAAb,CAAX,EAA0C,CAA1C,C;MAEIE,IAAI,6BAAJ,C;QACkB,yB;oBAAd,OAAC,cAAAd,C;UAAc,yB;UACmB,sB;;Yb7sBrB,U;YAAA,Sa6sBa,Ob7sBb,W;YAAhB,OAAGB,gBAAhB,C;cAAGB,2B;cAAM,Ia6sBgC,cb7sBIB,Oa6sBkB,EAAC,CAAd,sBb7sBIB,Oa6sBmD,OAAjC,ab7sBhC,C;gBAAwB,qBAAO,O;gBAAP,uB;;YAC9C,qBAAO,I;;;Ua4sBC,uC;UACA,IAAI,sBAAJ,C;YACI,OAAO,YAAS,cAAT,C;;;QAGD,2B;oBAAd,OAAC,gBAAd,C;UAAc,2B;UACmB,wB;;YbntBrB,U;YAAA,SamtBa,ObntBb,W;YAAhB,OAAGB,gBAAhB,C;cAAGB,6B;cAAM,IamtBgC,kBbntBIB,SamtBkB,EAaKB,CAAlB,sBbntBIB,SamtBuD,OAARc,abntBhC,C;gBAAwB,uBAAO,S;gBAAP,uB;;YAC9C,uBAAO,I;;;UaktBC,2C;UACA,IAAI,wBAAJ,C;YACI,OAAO,YAAS,gBAAT,C;;;MAlnB,OAAO,I;K;IAGX,iE;MAY+D,0B;QAAA,aAaKB,C;MAAG,0B;QAAA,aAAsB,K;MACtG,4BAAU,OAAV,EAAmB,UAAAnB,EAA+B,UAA/B,EAaKD,KAAID,C;K;IAEJ,mE;MAYmE,0B;QAAA,aAaKB,2B;MAAW,0B;QAAA,aAAsB,K;MACIH,4BAAU,OAAV,EAAmB,UAAAnB,EAA+B,UAA/B,EAaKD,IAAID,C;K;IAEJ,KE;MAWgE,0B;QAAA,aAaKB,C;MAAG,0B;QAAA,aAAsB,K;MACvG,gB;MAAA,8CAAU,OAAV,EAAmB,UAAAnB,EAA+B,UAA/B,EAaKD,KAAID,mDAAMe,E;K;IAEvE,sE;MAYoE,0B;QAAA,aAaKB,2B;MAAW,0B;QAAA,aAAsB,K;MACnH,gB;MAAA,8CAAU,OAAV,EAAmB,UAAAnB,EAA+B,UAA/B,EAaKD,IAAID,mDAaKE,E;K;IAKtE,6D;MAM4C,0B;QAAA,aAaKB,C;MAAG,0B;QAAA,aAAsB,K;MACnF,OAAW,cAAc,gCAAZB,GACI,sBAAW,mBAAy,IAAZ,CAAX,EAA8B,UAA9B,EAA0C,UAA1C,CADJ,GNz2B4F,kB6G3ME,oBvGujC5E,IuGvjC4E,C7G2MF,EM42BpE,UN52BoE,C;K;IM+2BhG,+D;MAQgD,0B;QAAA,aAaKB,C;MAAG,0B;QAAA,aAAsB,K;MACvF,OAAW,cAAc,gCAAZB,GACI,sBAAQ,MAAR,EAAGB,UAAhB,EAA4B,gBAA5B,EAAoC,UAApC,CADJ,GNx3B4F,kBM23B1E,MN33B0E,EM23BIE,UN33BkE,C;K;IM83BhG,iE;MAQgD,0B;QAAA,aAaKB,2B;MAAW,0B;QAAA,aAAsB,K;MAC/F,OAAW,cAAc,gCAAZB,GACI,0BAAe,mBAAy,IAAZ,CAAf,EAaKC,UAAIC,EAA8C,UAA9C,CADJ,GNp4BgG,sB6G3MM,oBvGklChF,IuGllCgF,C7G2MN,EMu4BpE,UNv4BoE,C;K;IM04BpG,mE;MAQoD,0B;QAAA,aAaKB,2B;MAAW,0B;QAAA,aAAsB,K;MACnG,OAAW,cAAc,gCAAZB,GACI,sBAAQ,MAAR,EAAGB,UAAhB,EAA4B,CAA5B,EAA+B,UAA/B,EAaKD,IAAID,CADJ,GNn5BgG,sBMs5B1E,MNt5B0E,EMs5BIE,UNt5BkE,C;K;IMy5BpG,mD;MAM+D,0B;QAAA,aAAsB,K;MACjF,OAAI,yBAAJ,GACI,sBAAQ,KAAR,UAA4B,UAA5B,KAA2C,CAD/C,GAGI,sBAAQ,KAAR,EAae,CAAf,EAaKB,gBAAIB,EAA0B,UAA1B,KAAyC,C;K;IAIjD,kD;MAMsD,0B;QAAA,aAAsB,K;MACxE,6BAAQ,IAAR,UAA2B,UAA3B,KAA0C,C;K;kFAE9C,4B;MAI0E,OAAA,KAAM,yBAAGB,SAAhB,C;K;IAM3C,yE;MACjC,oB;MACA,8B;MACA,oB;MACA,kC;K;IAG8C,sF;MAAA,gE;MAC1C,iBAAqB,E;MACrB,yBAAwC,WAAx,yCAAW,EAAS,CAAT,EAAY,oCAAM,OAAIB,C;MACxC,uBAA2B,sB;MAC3B,gBAA0B,I;MAC1B,eAAmB,C;K;0EAEEnB,Y;MACI,IAAI,uBAAKB,CAAtB,C;QACI,iBAAy,C;QACZ,gBAAW,I;;QAEX,IAAI,4CAAQ,CAAR,IAAa,uDAaA,yCAA1B,IAAmC,uBAAKB,yCAAM,OAA/D,C;UACI,gBAAW,qCAAyB,iBAAN,yCAAM,CAAZB,C;UACX,uBAAKB,E;;UAElB,YAAKB,iDAAN,yCAAM,EAAa,oBAAb,C;UACIB,IAAI,SAAS,IAAb,C;YACI,gBAAW,qCAAyB,iBAAN,yCAAM,CAAZB,C;YACX,uBAAKB,E;;YAEIB,IAAK,QAAiB,KAAjB,aAAL,EAAY,SAAU,KAAV,a;YACZ,gBAAW,gCAAWB,KAAxB,C;YACX,yBAAoB,QAAQ,MAAR,I;YACpB,uBAAKB,0BAAwB,WAAU,CAAd,GAAiB,CAAjB,GAAwB,CAA5C,K;;;QAG1B,iBAAy,C;K;oEAIpB,Y;MAKiB,Q;MAJb,IAAI,mBAAa,EAajB,C;QACI,iB;MACJ,IAAI,mBAAa,CAAjB,C;QACI,MAAM,6B;MACV,aAAa,mE;MAEb,gBAAW,I;MACX,iBAAy,E;MACZ,OAAO,M;K;uEAGX,Y;MACI,IAAI,mBAAa,EAajB,C;QACI,iB;MACJ,OAAO,mBAAa,C;K;;;IDA9C5B,Y;MAA8C,+D;K;;IAGEU,0E;MAAA,0C;QhB1mCjD,SgB2mCH,sBAAW,kBAAX,EAauB,YAAvB,EAaKD,kBAAID,C;QAAA,OAAwE,KAAK,CAAT,GAAY,IAAZ,GAAsB,OAAM,CAAN,C;O;K;IAdIG,iF;MAUkE,0B;QAAA,aAaKB,C;MAAG,0B;QAAA,aAAsB,K;MAAO,qB;QAAA,QAAa,C;MAC7H,wBAAwB,KAAxB,C;MAEA,OAAO,4BAAwB,SAAXB,EAA8B,UAA9B,EAA0C,KAA1C,EAAiD,gDAAjD,C;K;IAwBiD,gF;MAAA,0C;QAAKB,Q;QAAA,oCAAU,sBAAV,EAA0B,YAA1B,EAAqD,kBAArD,EAAwE,KAAxE,aAAsF,GAAG,UAAH,EAAe,WAAO,OAAtB,CAAtF,O;O;K;IAIB9E,mF;MAC0E,0B;QAAA,aAaKB,C;MAAG,0B;QAAA,aAAsB,K;MAAO,qB;QAAA,QAAa,C;MACrI,wBAAwB,KAAxB,C;MACA,qBAAgC,OAAX,UAAW,C;MAEHc,OAAO,4BAAwB,SAAXB,EAA8B,UAA9B,EAA0C,KAA1C,EAAiD,sDAAjD,C;K;IAIX,wC;MnBltCI,IAAI,EmBmtCI,SAAS,CnBntCb,CAAJ,C;QACI,cmBktCkB,8C;QnBjtCIB,MAAM,gCAAyB,OAAQ,WAAjC,C;Q;ImBkuCgE,sD;MAAA,qB;QAAE,yCAAU,EAAY,C;O;K;IAZhF,mE;MAWmE,0B;QAAA,aAAsB,K;MAAO,qB;QAAA,QAAa,C;MACzG,OAAsE,OAAtE,+BAaKB,UAAIB,UAA2C,UAA3C,EAA+D,KAA/D,CAAsE,EAAL,iCAAJ,C;K;IAE1E,yD;MAWyD,0B;QAAA,aAAsB,K;MAAO,qB;QAAA,QAAa,C;MAC/F,IAAI,UAAW,OAAX,KAAmB,CAAvB,C;QACI,gBAAgB,WAAW,CAAX,C;QACHb,IAAI,EAA

C,SAh/BuC,YAAU,CAg/BID,CAAJ,C;UACI,OAAO,mBAAM,SAAN,EAAiB,UAAjB,EAA6B,KAA7B,C;UAI2E,kBAAb,cAAtE,+BAAkB,UAAIB,UAA2C,UAA3C,EAA+D,KAA/D,CAAsE,C;Mb80tE,kBAAM,iBAAa,qCAAwB,EAAxB,CAAb,C;MAuEA,Q;MAAA,6B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,WAAY,WatTgF,uBbsTIE,IatTke,CbsThF,C;;MatThB,ObuTO,W;K;Ia5SmE,wD;MAAA,qB;QAAE,yCAAU,EAAV,C;O;K;IARhF,qE;MAOiE,0B;QAAA,aAAsB,K;MAAO,qB;QAAA,QAAa,C;MACvG,OAASe,OAAtE,6BAAkB,UAAIB,UAA2C,UAA3C,EA A+D,KAA/D,CAAsE,EAAl,mCAAJ,C;K;IAE1E,2D;MAOuD,0B;QAAA,aAAsB,K;MAAO,qB;QAAA,QAAa,C;MAC7F,IAAI,UAAW,OAAx,KAAmB,CAAvB,C;QACI,OAAO,mBAAoB,oBAAd,WAaw,CAAX,CAAc,CAApB,EAAgC,UAAhC,EAA4C,KAA5C,C;OAG+E,kBAAb,cAAtE,6BAAkB,UAAIB,UAA2C,UAA3C,EAA+D,KAA/D,CAAsE,C;MbqNtE,kBAAM,iBAAa,qCAAwB,EAAxB,CAAb,C;MAuEA,Q;MAAA,6B;MAAb,OAAa,cAAb,C;QAAa,sB;QACT,WAAY,Wa7RgF,uBb6RlE,Ia7RkE,Cb6RhF,C;;Ma7RhB,Ob8RO,W;K;Ia3RX,0D;MASI,wBAAwB,KAAxB,C;MAEA,oBAAoB,C;MACpB,gBAAgB,sBAAQ,SAAR,EAAMB,aAAnB,EAakC,UAAIC,C;MACHb,IAAI,cAAa,EAAb,IAAmB,UAAAS,CAAhC,C;QACI,OAAO,OAAO,SAAK,WAAZ,C;OAGX,gBAAgB,QAAQ,C;MACxB,aAAa,iBAAsB,SAAJ,GAAqB,eAAN,KAAM,EAAa,EAAb,CAArB,GAA2C,EAA7D,C;;QAET,MAAO,WA36B6E,8BA26B/D,aA36B+D,EA26BhD,SA36BgD,CAAKC,WA26B/G,C;QACP,gBAAgB,YAAY,SAAU,OAAtB,I;QAEhB,IAAI,aAAa,MAAO,KAAP,MAAe,QAAQ,CAAR,IAAf,CAAJB,C;UAA2C,K;QAC3C,YAAY,sBAAQ,SAAR,EAAMB,aAAnB,EAakC,UAAIC,C;;MACP,sBAAa,EAAb,C;MAET,MAAO,WAI7BiF,8BAk7BnE,aAI7BmE,EAk7BpD,gBAI7BoD,CAAKC,Wak7BnH,C;MACP,OAAO,M;K;2EAGX,mC;MAOmD,qB;QAAA,QAAa,C;MAAmB,OAAA,KAAM,eAAM,SAAN,EAAY,KAAZ,C;K;+FAEzF,mC;MAU6D,qB;QAAA,QAAa,C;MAAuB,OA AA,KAAM,yBAAGB,SAAhB,EAAsB,KAAtB,C;K;IAEvG,iC;MAK2D,mCAAGB,MAAhB,EAawB,IAAxB,EAA8B,IAA9B,E;K;IAE3D,0B;MAKgD,OAAe,UAAf,uBA Ae,C;K;IAqB/D,uD;MAQsB,Q;MAPIB,IAAI,iCAAKB,yBAAtB,C;QACI,OAAy,SAAL,SAAK,EAAO,KAAP,EAA2B,IAA3B,C;OAGhB,IAAI,cAAS,KAAb,C;QAAoB,OAA O,I;MAC3B,IAAI,qBAAGB,aAhB,IAAiC,SAAK,OAAL,KAAe,KAAM,OAAID,C;QAAkE,OAAO,K;MAEvD,u B;MAAIB,aAAU,CAAV,gB;QACI,IAAI,CAAS,SAAR,qBAAK,CAAL,CAAQ,EAAO,iBAAM,CAAN,CAAP,EA A8B,IAA9B,CAAb,C;UACI,OAAO,K;;MAIf,OAAO,I;K;IAGX,6C;MAQsB,Q;MAPIB,IAAI,iCAAKB,yBAAtB,C; QACI,OAAO,kBAAQ,KAAR,C;OAGX,IAAI,cAAS,KAAb,C;QAAoB,OAAO,I;MAC3B,IAAI,qBAAGB,aAhB,IA AiC,SAAK,OAAL,KAAe,KAAM,OAAID,C;QAAkE,OAAO,K;MAEvD,uB;MAAIB,aAAU,CAAV,gB;QACI,IA AI,qBAAK,CAAL,MAAW,iBAAM,CAAN,CAAF,C;UACI,OAAO,K;;MAIf,OAAO,I;K;IAGX,oC;MAU+C,QA AM,SAAN,C;aAC3C,M;UAD2C,OACjC,I;aACV,O;UAF2C,OAehC,K;gBACH,MAAM,gCAAyB,mDAAGD,SA AzE,C;;K;IAGIB,0C;MAUsD,QAAM,SAAN,C;aACID,M;UADkD,OACxC,I;aACV,O;UAFkD,OAevC,K;gBAFu C,OAG1C,I;;K;I8Kr8CZ,sB;MAAA,0B;MAII,aAC+B,e;MAC/B,cACgC,e;MACHc,WAC6B,e;MAC7B,YAC8B,e; MAC9B,eACiC,e;MACjC,YAC8B,gB;MAC9B,aAC+B,gB;MAC/B,YAC8B,gB;MAC9B,aAC+B,gB;MAC/B,eAC iC,gB;MACjC,iBACmC,gB;MACnC,qBAEuC,gB;MACvC,sBAEwC,gB;MACxC,kBACoC,gB;MACpC,cACgC,g B;MACHc,iBACmC,gB;MACnC,iBACmC,gB;MACnC,iBACmC,gB;MACnC,YAC8B,gB;MAC9B,aAC+B,iB;M AC/B,aAC+B,iB;MAC/B,uBACyC,iB;MACzC,wBAC0C,iB;MAC1C,sBACwC,iB;MACxC,uBACyC,iB;MACzC, wBAC0C,iB;MAC1C,sBACwC,iB;MACxC,cACgC,iB;MACHc,oBACsC,iB;MACTc,cACgC,iB;MACHc,gBACKC ,iB;MACIC,aAC+B,iB;MAC/B,mBACqC,iB;MACrC,YAC8B,iB;MAC9B,UAC4B,iB;MAC5B,mBACqC,iB;MAC rC,gBACKC,iB;MACIC,mBACqC,iB;MACrC,sBACwC,iB;MAExC,sBAGwC,gB;MAExC,uBAGyC,gB;K;;;IA7F7 C,kC;MAAA,iC;QAAA,gB;OAAA,0B;K;;;;2FCuE0C,Y;MAAQ,oCAAa,IAAb,C;K;IAiBpB,yC;MAAqB,kB;K; mIAC3C,Y;MACmD,OAAA,UAAM,YAAN,aAAkB,CAAIB,C;K;mIACnD,Y;MACmD,OAAA,UAAM,YAAN,a AAkB,CAAIB,C;K;mIACnD,Y;MACmD,OAAA,UAAM,YAAN,aAAkB,CAAIB,C;K;mIACnD,Y;MACmD,OAAA,UA AM,YAAN,aAAkB,CAAIB,C;K;mIACnD,Y;MACmD,OAAA,UAAM,YAAN,aAAkB,CAAIB,C;K;mIACnD,Y;MACmD,OAAA,UAAM,YAAN,aAAkB,CAAIB,C;K;mIACnD,Y;MACmD,OAAA,UAAM,YAAN,aAAkB,CAAIB,C;K;mIACnD,Y;MACmD,OAAA,UAAM,YAAN,aAAkB,CAAIB,C;K;qIACnD,Y;MACmD,OAAA,UAAM,YAAN,aAAkB,CAAIB,C;K;gDAEnD,Y;MA MoC,OAAA,UAAM,YAAY,iBAAQ,CAAR,EAaw,UAAM,YAAY,KAA7B,C;K;;;6EhEjH9D,yB;MAAA,iD;MA AA,4B;QAI4C,kBAAM,SAAN,C;O;KAJ5C,C;+EAMA,yB;MAAA,gD;MAAA,oC;QAI+D,kBAAM,SAAN,EAAY ,MAAZ,C;O;KAJ/D,C;+EAMA,yB;MAAA,oC;MAAA,qC;QAIqE,sBAAM,SAAN,EAAY,OAAZ,C;O;KAJrE,C;Ii Y4B,4B;MAmBxB,gC;MANB6C,0B;MAW7B,UAEA,MAFA,EAGA,M;MALZ,IiJc8D,IjIc9D,C;QACI,IAAI,kB

AAJ,C;UACQ,mB;UAAJ,IAAI,sEAA sB,SAA tB,EAAJ,C;YAAqC,MAAM,sBAAiB,YAAF,+CAAf,C;;UAEvC,qB;  
UAAJ,IAAI,0EAAuB,UAAvB,EAAJ,C;YAAuC,MAAM,sBAAiB,YAAF,gDAAF,C;UACzC,qB;UAAJ,IAAI,kEA  
A+B,mBAA/B,CAAJ,C;YAAwD,MAAM,sBAAiB,YAAF,mCAAf,C;;Q:mFAZID,Y;MAAQ,kCAAa,CAAb,C;K;+  
FACU,Y;MAAQ,OAAA,eAAS,QAAT,GAAqB,C;K;qCACvE,Y;MAA0B,QADwB,eAAS,QAAT,GAAqB,CAC7C  
,MAAqB,C;K;sCAC/C,Y;MAA2B,QAFuB,eAAS,QAAT,GAAqB,CAE5C,MAAqB,C;K;yFACxB,Y;MAAQ,OAA  
I,kBAAJ,mF;K;IAAhC,8B;MAAA,kC;MACI,YAC4B,gB;MAE5B,gBACgC,iBAAiB,UAAjB,C;MACHc,4BAAsC,  
uC;K;mDAEtC,yC;MAGI,2BAAoB,KAApB,EAA2B,UAA3B,EAAuC,UAAvC,C;K;iJAM8B,yB;MAAA,6C;MAA  
A,iD;MAAA,4B;QAAQ,sD;O;KAAR,C;iJAIC,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,sD;O;KAAR,C;iJAU  
E,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,sD;O;KAAR,C;mJAKF,yB;MAAA,6C;MAAA,iD;MAAA,4B;QA  
AQ,uD;O;KAAR,C;mJAIC,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,uD;O;KAAR,C;mJAUE,yB;MAAA,6C;  
MAAA,iD;MAAA,4B;QAAQ,uD;O;KAAR,C;mJAKH,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,uD;O;KAAR  
,C;mJAIC,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,uD;O;KAAR,C;mJAUE,yB;MAAA,6C;MAAA,iD;MAA  
A,4B;QAAQ,uD;O;KAAR,C;yIAKR,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,kD;O;KAAR,C;yIAIC,yB;MA  
AA,6C;MAAA,iD;MAAA,4B;QAAQ,kD;O;KAAR,C;yIAUE,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,kD;O;  
KAAR,C;yIAKH,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,kD;O;KAAR,C;yIAIC,yB;MAAA,6C;MAAA,iD;  
MAAA,4B;QAAQ,kD;O;KAAR,C;yIAUE,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,kD;O;KAAR,C;qIAKL,y  
B;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,gD;O;KAAR,C;qIAIC,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,g  
D;O;KAAR,C;qIAUE,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,gD;O;KAAR,C;mIAKJ,yB;MAAA,6C;MAA  
A,iD;MAAA,4B;QAAQ,+C;O;KAAR,C;mIAIC,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,+C;O;KAAR,C;mIA  
UE,yB;MAAA,6C;MAAA,iD;MAAA,4B;QAAQ,+C;O;KAAR,C;uDAK9B,iB;MAK+C,OAAM,WAAN,KAAM,y  
C;K;uDAErD,iB;MAKgD,OAAM,aAAN,KAAM,yC;K;uDAEtD,iB;MASkD,OAAM,aAAN,KAAM,yC;K;wDAGx  
D,iB;MAKgD,OAAM,WAAN,KAAM,0C;K;wDAEtD,iB;MAKiD,OAAM,aAAN,KAAM,0C;K;wDAEvD,iB;MA  
SmD,OAAM,aAAN,KAAM,0C;K;wDAGzD,iB;MAKgD,OAAM,WAAN,KAAM,0C;K;wDAEtD,iB;MAKiD,OA  
AM,aAAN,KAAM,0C;K;wDAEvD,iB;MASmD,OAAM,aAAN,KAAM,0C;K;mDAGzD,iB;MAK2C,OAAM,WA  
AN,KAAM,qC;K;mDAEjD,iB;MAK4C,OAAM,aAAN,KAAM,qC;K;mDAEID,iB;MAS8C,OAAM,aAAN,KAAM,  
qC;K;mDAGpD,iB;MAK2C,OAAM,WAAN,KAAM,qC;K;mDAEjD,iB;MAK4C,OAAM,aAAN,KAAM,qC;K;mD  
AEID,iB;MAS8C,OAAM,aAAN,KAAM,qC;K;iDAGpD,iB;MAKyC,OAAM,WAAN,KAAM,mC;K;iDAE/C,iB;M  
AK0C,OAAM,aAAN,KAAM,mC;K;iDAEhD,iB;MAS4C,OAAM,aAAN,KAAM,mC;K;gDAGID,iB;MAKwC,OA  
AM,WAAN,KAAM,kC;K;gDAE9C,iB;MAKyC,OAAM,aAAN,KAAM,kC;K;gDAE/C,iB;MAS2C,OAAM,aAAN,  
KAAM,kC;K;iDAEjD,iB;;QAY4C,OACx C,cAAc,KAA d,EAAiC,KAAjC,C;;QACF,+C;UACE,MAAM,6BAAyB,s  
CAAmC,KAA nC,OAAzB,EAA sE,CAAtE,C;;UAHkC,O;;K;0DAM5C,iB;;QAMqD,OACjD,cAAc,KAA d,EAAiC,I  
AAjC,C;;QACF,+C;UACE,MAAM,6BAAyB,0CAAuC,KAAvC,OAAzB,EAA0E,CAA1E,C;;UAH2C,O;;K;uDAM  
rD,iB;;QAWmD,OAC/C,cAAc,KAA d,EAAiC,KAAjC,C;;QACF,+C;UAFiD,OAG/C,I;;UAH+C,O;;K;gEAMnD,iB  
;;QAK4D,OACxD,cAAc,KAA d,EAAiC,IAAjC,C;;QACF,+C;UAF0D,OAGxD,I;;UAHwD,O;;K;;IA/XhE,0C;MA  
AA,yC;QAAA,wB;OAAA,kC;K;oCAwYA,Y;MAC6C,kBAAy,YAAD,aAAX,EAzZK,eAAS,QAAT,GAAqB,CAY  
Z1B,C;K;qCAE7C,iB;MAiBW,Q;MATH,IAAA,IAAK,aAAL,C;QACI,IAAI,KAAM,WAAN,IAAqB,IAAK,WAA  
L,KAAkB,KAAM,WAAxB,gBAAoC,CAA7D,C;UACI,OAAO,I;;UAEP,MAAM,gCAAyB,2EAAzB,C;WAE d,IA  
AA,KAAM,aAAN,C;QAAsB,OAAO,K;MAI7B,KA7a0C,eAAS,QAAT,GAAqB,CA6a/D,OAA0B,KA7agB,WAAS  
,QAAT,GAAqB,CA6a/D,E;QACI,aAAa,IAAK,QAAL,KAAa,KAAM,QAAnB,C;QAET,uB;UACI,iCAA0B,MAA1  
B,C;;UAEA,kCAA2B,MAA3B,C;aAGZ,IAAA,IAAK,eAAL,C;QACI,mCAAqB,IAAK,QAA1B,EAAiC,KAAM,Q  
AAvC,C;;QAEA,mCAAqB,KAAM,QAA3B,EAAkC,IAAK,QAAvC,C;MabR,W;K;gDAiBJ,kC;MAGW,Q;MAFP,  
kBAAkB,cAAc,UAA d,C;MACIB,mBAAmB,eAAa,WAAb,C;MACZ,IAAI,8EAA sC,mBAA tC,CAAJ,C;QACH,yB  
AAyB,oBAAa,cAAc,WAA d,CAAb,C;QACzB,uBAAgB,cAAc,YAA d,MAA8B,kBAA9B,CAAhB,C;;QAEA,wBA  
A8B,WAAb,YAAa,yBAAsB,UAA tB,CAA9B,C;;MAJJ,W;K;sCAQJ,iB;MAMuD,wBAAS,KAAD,aAAR,C;K;uCA  
EvD,iB;MAQe,UAUJ,M;MAXP,IAAI,iBAAJ,C;QAEQ,cAAS,CAAT,C;UAAc,MAAM,gCAAyB,mEAAzB,C;aAC  
pB,YAAQ,CAAR,C;UAAa,W;;UACL,OAAC,IAAD,a;QAHZ,W;OAMJ,IAAI,UAA s,CAAb,C;QAAgB,OAAO,qC  
;MAEvB,YAA Y,Y;MACZ,aAAa,mCAAQ,KAAR,E;MACN,IAAI,kBAAJ,C;QACH,IAAI,yEAAJ,C;UAEI,yBAAg  
B,MAAhB,C;;UAEA,IAAI,sCAAS,KAAT,IAAkB,KAAIB,CAAJ,C;YACI,mCAA0B,MAA1B,C;;YAEA,aAAa,cA

Ac,KAAd,C;YACb,eAAe,eAAQ,cAAc,MAAd,CAAR,C;YACf,mBAAmB,oCAAS,KAAT,E;YACnB,kBAAkB,iB  
AAe,cAAc,sCAAW,KAAX,EAAd,CAAf,C;YACIB,IAAI,4CAAe,KAaf,IAAwB,MAAxB,KAakC,gBAAgB,YAA  
hB,gBAAgC,CAAT,E,C;cACI,0BAA6B,WAAZ,WAAY,EAAS,8BAAa,UAAb,CAAT,CAA7B,C;;cAEA,SAAI,YA  
AM,WAAN,KAAM,CAAN,EAAMB,WAAN,KAAM,CAAnB,IAA0B,CAA9B,GAAiC,yCAAjC,GAA+C,qD;;;;;Q  
AK3D,IAAI,sCAAS,KAAT,IAAkB,KAAiB,CAAJ,C;UACI,0BAAwB,WAAP,MAAO,EAAS,8BAAa,UAAb,CAA  
T,CAAxB,C;;UAEA,SAAI,YAAM,WAAN,KAAM,CAAN,EAAMB,WAAN,KAAM,CAAnB,IAA0B,CAA9B,GA  
AiC,yCAAjC,GAA+C,qD;;;MAvBvD,a;K;uCA4BJ,iB;MASI,eAAqB,WAAN,KAAM,C;MACrB,IAAa,QAAT,KA  
AuB,KAA3B,C;QACI,OAAO,mBAAM,QAAN,C;OAGX,WAAW,kB;MACX,aAAa,sBAAS,IAAT,IAAiB,K;MA  
C9B,OAAc,aAAP,MAAO,EAAW,IAAX,C;K;qCAGiB,iB;MAQe,Q;MADX,IAAI,UAAS,CAAb,C;QAEQ,sB;UA  
AgB,gD;aAChB,sB;UAAgB,4D;;UACR,MAAM,gCAAYB,4DAzB,C;QAHIB,W;OAMJ,IAAI,kBAAJ,C;QACI,O  
AAO,gBAAgB,qCAAQ,KAAR,EAAhB,C;;QAEP,IAAI,iBAAJ,C;UACI,OAAO,mBAAa,WAAN,KAAM,CAAb,C  
;QAEX,aAAa,qCAAQ,KAAR,E;QAEb,IAAI,kEAAGC,mBAAhC,CAAJ,C;UACI,UAAU,cAAc,sBAAS,oCAAS,K  
AAT,EAAT,CAAd,0BAA0C,KAA1C,E;UACV,OAAO,gBAAgB,cAAc,MAAd,MAAwB,GAAXB,CAAhB,C;SAE  
X,OAAO,iBAAiB,MAAjB,C;;K;qCAIf,iB;MAOI,eAAqB,WAAN,KAAM,C;MACrB,IAAa,QAAT,KAAuB,KAAv  
B,IAAgC,aAAY,CAAhD,C;QACI,OAAO,iBAAI,QA AJ,C;OAGX,WAAW,kB;MACX,aAAa,sBAAS,IAAT,IAAiB  
,K;MAC9B,OAAc,aAAP,MAAO,EAAW,IAAX,C;K;oCAGiB,iB;MAEI,kBAAkB,SAAM,IAAK,cAAX,EAAWB,K  
AAM,cAA9B,C;MACiB,OAAO,IAAK,kBAAS,WAAT,CAAL,GAA6B,KAAM,kBAAS,WAAT,C;K;oCAG9C,Y;  
MACmC,oCAAW,C;K;oCAE9C,Y;MACmC,oCAAW,C;K;oCAE9C,Y;MACmC,+BAAY,yCAAS,WAARb,KAAi  
C,wBAAY,qDAAa,WAAzB,C;K;kCAEpE,Y;MACiC,QAAC,iB;K;yFAGC,Y;MAAQ,OAAI,iBAAJ,GAAMb,IAA  
D,aAAlB,GAA6B,I;K;yCAExE,iB;MACI,kBAAkB,IAAK,WAAL,KAakB,KAAM,WAAxB,C;MACiB,IAAI,yBA  
Ac,CAAd,IAAMb,CAAA,WAAY,QAAZ,GAAwB,CAAxB,MAA6B,CAApD,C;QACI,OAAO,IAAK,WAAS,iBA  
AU,KAAM,WAAhB,C;MAEzB,QAAQ,CA11BsC,eAAS,QAAT,GAAqB,CA01B3D,KAAyB,KA11Ba,WAAS,QAA  
T,GAAqB,CA01B3D,K;MACR,OAAW,iBAAJ,GAakB,CAAC,CAAD,IAAiB,GAA0B,C;K;uHAMrC,kB;MAeI,O  
AAO,OAAO,gBAAP,EAAoB,mBAAPB,EAAoC,qBAAPC,EAAsD,qBAAtD,EAawE,yBAAXE,C;K;uHAGX,kB;  
MAcI,OAAO,OAAO,iBAAP,EAAqB,qBAARb,EAAuC,qBAAvC,EAAYD,yBAAZD,C;K;uHAGX,kB;MAaI,OAA  
O,OAAO,mBAAP,EAAuB,qBAAvB,EAAYC,yBAAZC,C;K;uHAGX,kB;MAYI,OAAO,OAAO,mBAAP,EAAuB,y  
BAAvB,C;K;0FAKP,Y;MAAQ,OAAI,iBAAJ,GAakB,CAAIB,GAA0B,6CAAe,EAaf,EAAMb,Q;K;4FAIrD,Y;M  
AAQ,OAAI,iBAAJ,GAakB,CAAIB,GAA0B,+CAAiB,EAAjB,EAAqB,Q;K;4FAIvD,Y;MAAQ,OAAI,iBAAJ,GA  
AkB,CAAIB,GAA0B,+CAAiB,EAAjB,EAAqB,Q;K;gGAIvD,Y;MACI,sB;QADI,OACY,C;WACHb,wB;QAFI,OA  
EY,cAAc,wCAAQ,IAAR,EAAd,CAA6B,Q;;QAFzC,OAGK,wCAAQ,UAAR,EAAuB,Q;K;0CAMxC,gB;MAQiB,  
UAAN,M;MAAM,sB;MACT,iBAAA,yCAAS,WAAT,E;QAA4B,SAAP,wCAAO,kB;WAC5B,iBAAA,qDAAa,W  
AAb,E;QAAgC,SAAP,wCAAO,kB;;QAG5B,6BAAoB,YAAM,WAA1B,EAAsC,kBAAtC,EAAMd,IAAnD,C;;MA  
LR,a;K;wCAUJ,gB;MAUiB,UAAN,M;MAAM,sB;MACT,iBAAA,yCAAS,WAAT,E;;WACA,iBAAA,qDAAa,W  
AAb,E;;;QACQ,+BAAoB,YAApB,EAA2B,kBAA3B,EAAwC,IAAxC,C;MAHZ,a;K;uCAOJ,gB;MAUI,OAAa,WA  
Ab,oBAAO,IAAP,CAAA,4BAAyD,Q;K;kFAKhD,Y;MAAQ,6D;K;mFAKP,Y;MAAQ,8D;K;qFAKN,Y;MAAQ,gE  
;K;qFAKR,Y;MAAQ,gE;K;0FAKH,Y;MAAQ,qE;K;0FAKR,Y;MAAQ,qE;K;yFAKT,Y;MAAQ,oE;K;uFASrC,Y;  
MAAQ,2D;K;wFAQR,Y;MAAQ,4D;K;0FAQR,Y;MAAQ,8D;K;0FAQR,Y;MAAQ,8D;K;+FAQR,Y;MACI,OAA  
W,uBAAGB,eAApB,GAAGC,YAAhC,GAA2C,4D;K;+FAatD,Y;MAAQ,mE;K;8FAYR,Y;MAEW,Q;MADP,YAA  
Y,Y;MAER,uB;QAae,Y;WACf,8C;;WACA,+C;;;QACQ,qBAAc,KAAd,C;MAJZ,W;K;2CAUR,Y;MASuC,8B;K;  
4CAEvC,Y;MASwC,+B;K;kCAExC,Y;MAuBwC,Q;MAAA,sB;MACpC,qB;QAD8B,OACxB,I;WACN,iBAAA,y  
CAAS,WAAT,E;QAF8B,OAET,U;WACrB,iBAAA,qDAAa,WAAb,E;QAH8B,OAGL,W;;QAErB,iBAAiB,iB;Q6  
HzhBF,gBAAhB,sB;Q7H2hBK,e;UAAgB,yBAAO,EAAP,C;QACF,YAAAd,kB;QA9RD,WAAO,iB;QAAP,YAAoB,  
oB;QAAPB,cAAoC,sB;QAAPC,cAAsD,sB;QAAtD,kBAAwE,0B;QAS/D,0B;QAPI,cAAc,iB;QACd,eAAe,UAAS  
,C;QACxB,iBAAiB,YAAW,C;QAC5B,iBAAiB,YAAW,CAAX,IAAGB,gBAAE,C;QACHD,iBAAiB,C;QACjB,IA  
AI,OAAJ,C;UACI,yBAAO,IAAP,CAAA,gBAAO,GAAP,C;UACb,+B;SAEJ,IAAI,aAAa,YAAY,cAAc,UAA1B,CA  
Ab,CAAJ,C;UACI,IAAI,6DAAe,CAAnB,C;YAAsB,yBAAO,EAAP,C;UACtB,yBAAO,KAAP,CAAc,gBAAO,GA  
AP,C;SAEIB,IAAI,eAAe,eAAe,YAAY,OAA3B,CAAf,CAAJ,C;UACI,IAAI,6DAAe,CAAnB,C;YAAsB,yBAAO,E  
AAP,C;UActB,yBAAO,OAAP,CAAGB,gBAAO,GAAP,C;SAEpB,IAAI,UAAJ,C;UACI,IAAI,6DAAe,CAAnB,C;

YAAsB,yBAAO,EAAP,C;UAEIB,gBAAW,CAAX,IAAgB,OAAhB,IAA2B,QAA3B,IAAuC,UAAvC,C;YACI,mC  
AAiB,OAAjB,EAA0B,WAA1B,EAAuC,CAAvC,EAA0C,GAA1C,EAA2D,KAA3D,C;eACJ,mBA Ae,OAAf,C;YA  
CI,mCAAiB,cAAc,OAAd,IAAjB,EAA0C,cAAc,OAAxD,EAAMe,CAAnE,EAAsE,IAAtE,EAAwF,KAAxF,C;eAC  
J,mBA Ae,IAAf,C;YACI,mCAAiB,cAAc,IAAd,IAAjB,EAAsC,cAAc,IAApD,EAA2D,CAA3D,EAA8D,IAA9D,EA  
AgF,KAAhF,C;;YAEA,yBAAO,WAAP,CAAoB,gBAAO,IAAP,C;SAGhC,IAAI,cAAc,aAAa,CAA/B,C;UAAkC,y  
BAAO,CAAP,EAAU,EA AV,CAAe,gBAAO,EAAP,C;QAvC/B,OOx1B3B,SsHoUqC,W;;K;4C7HikB5C,yE;MACI  
,yBAAO,KAAP,C;MACA,IAAI,eAAc,CAAIB,C;QACI,yBAAO,EAAP,C;QACA,iBA AuC,WAAiB,UAAW,WAA  
W,EAAS,cAAT,EAAYB,EA AzB,C;QACR,sB;;UsB5zBzB,Q;UAAA,OAAQ,WAAR,etB4zBc,UsB5zBd,CAAQ,C  
AAR,W;UAA d,OAAc,cAA d,C;YAAc,uB;YACV,ItB2zBiD,UsB3zBnC,YtB2zBU,UsB3zBV,YAAK,KAAL,EtB2z  
BmC,MAAM,EsB3zBvD,C;cACI,qBAAO,K;cAAP,uB;;UAGR,qBAAO,E;;;QtBuzBC,oBAAoB,qBA AuC,CAA vC  
,I;QAEhB,KAAC,SAAD,IAAc,gBA AgB,CAA9B,C;UAAmC,8BAAY,UAAZ,EA AwB,CAAxB,EAA2B,aAA3B,C;  
;UAC3B,8BAAY,UAAZ,EA AwB,CAAxB,EAA2B,CAAC,CAAC,gBA AgB,CAAhB,IAAD,IAAsB,CAAtB,IAAD,  
IAA4B,CAA5B,IAA3B,C;OAGhB,yBAAO,IAAP,C;K;0CAGJ,0B;MAGbWc,wB;QAAA,WAAgB,C;MIn9BxD,IA  
AI,EJo9BQ,YAAY,CiP9BpB,CAAJ,C;QACI,cJm9ByB,oD;QII9BzB,MAAM,gCAAYB,OAAQ,WAAjC,C;OJm9B  
N,aAAa,sBAAS,IAAT,C;MACb,IAAW,WAAP,MAAO,CAAX,C;QAAyB,OAAO,MAAO,W;MACvC,OAAO,sB  
AAsB,MAAtB,EAAuC,eAAT,QAAS,EAAa,EAAb,CAA vC,IAAgE,UAAAL,IAAK,C;K;qCAI3E,Y;M6HvmBuB,gB  
AAhB,sB;M7HqnBH,IAAI,iBA AJ,C;QAAkB,yBAAO,EAAP,C;MACIB,yBAAO,IAAP,C;MAC4B,YAA d,kB;MA  
xWP,YAAO,kB;MAAP,cAAqB,sB;MAArB,cAAuC,sB;MAAvC,kBAAYD,0B;MAYW5D,cACY,K;MACZ,IAAI,i  
BA AJ,C;QAEI,wB;OAEJ,eAAe,oB;MACf,iBA AiB,YAAW,CAAX,IAAgB,gBA Ae,C;MACHd,iBA AiB,YAAW,C  
AAX,KAAiB,cAAc,QAA/B,C;MACjB,IAAI,QAAJ,C;QACI,yBAAO,OAAQ,CAAc,gBAAO,EAAP,C;OAEIB,IAA  
I,UAAJ,C;QACI,yBAAO,OAAQ,CAAgB,gBAAO,EAAP,C;OAEpB,IAAI,eAAe,CAAC,QAAD,IAAa,CAAC,UAA  
7B,CAAJ,C;QACI,mCAAiB,OAAjB,EAA0B,WAA1B,EAAuC,CAA vC,EAA0C,GAA1C,EAA2D,IAA3D,C;OAp  
BuB,OOx7B5B,SsHoUqC,W;K;;;kC7H5YhD,Y;MAAA,c;MAuBiD,2D;MAvBjD,a;K;gCAAA,iB;MAAA,2IAuBi  
D,gDAvBjD,G;K;IA8hCA,qC;MAIW,Q;MAAA,IAAI,6DA AJ,C;QACH,uBA AgB,4BA AiC,oBAAL,SAAK,CAAj  
C,EAA2C,IAA3C,yCAAhB,C;;QAES,oBAAT,8BAAS,EA AW,IAAX,C;MAHb,W;K;IAMJ,uC;MAII,kBA AkB,4B  
AA4B,SAA5B,0CAAiE,IAAjE,C;MACIB,IAAa,WAAD,aAAR,yDAAsB,WAAiB,CAAJ,C;QACI,OAAO,gBA Ag  
B,4BAA4B,SAA5B,EA AkC,IAAIc,yCAAhB,C;;QAEp,aAAa,sBA AoB,SAAPB,EAA0B,IAA1B,0C;QACb,OAAO  
,iBA AwB,WAAP,MAAO,yBAAsB,UAAiB,CAAxB,C;;K;IAIf,uC;MAAw,Q;MAHP,gBA AgB,oBAAoB,SAAPB,E  
AA0B,IAA1B,yC;MIviChB,IAAI,CJwiCl,CAAW,QAAV,SAAU,CiXiCnB,C;QACI,cJuiC0B,+B;QItiC1B,MAAM,  
gCAAYB,OAAQ,WAAjC,C;OJuiCV,YAA sB,YAAV,SAAU,C;MACf,IAAI,sEAAqB,SAArB,CAAJ,C;QACH,uBA  
AgB,KAAhB,C;;QAEA,aAAwE,YAA3D,oBAAoB,SAAPB,EAA0B,IAA1B,0CAA2D,C;QACxE,kCAA2B,MAA3  
B,C;;MAJJ,W;K;IAgBuB,oC;MAAQ,oE;K;IAOP,sC;MAAQ,sE;K;IAWN,sC;MAAQ,sE;K;IAQV,qC;MAAQ,qE;K  
;IAOP,uC;MAAQ,uE;K;IAWN,uC;MAAQ,uE;K;IAQX,qC;MAAQ,qE;K;IAOP,uC;MAAQ,uE;K;IAWN,uC;MAA  
Q,uE;K;IAQhB,gC;MAAQ,gE;K;IAOP,kC;MAAQ,ke;K;IAWN,kC;MAAQ,ke;K;IAQX,gC;MAAQ,gE;K;IAOP,k  
C;MAAQ,ke;K;IAWN,kC;MAAQ,ke;K;IAQb,8B;MAAQ,8D;K;IAOP,gC;MAAQ,gE;K;IAWN,gC;MAAQ,gE;K;I  
AQZ,6B;MAAQ,6D;K;IAOP,+B;MAAQ,+D;K;IAWN,+B;MAAQ,+D;K;yEAG/B,+B;MAIqE,8BAAW,SAAX,C;  
K;2EAERe,+B;MAUwE,8BAAW,SAAX,C;K;IAIxE,yC;MACI,aAAa,KAAM,O;MACnB,IAAI,WAAU,CAAd,C;Q  
AAiB,MAAM,gCAAYB,qBAAzB,C;MACvB,YAAY,C;MACZ,aAAa,gCAAS,K;MACtB,qBAAqB,U;MACrB,QA  
AM,iBAAM,KAAN,CAAN,C;aACI,E;aAAA,E;UAAy,qB;UAAZ,K;;MAEJ,cAAc,QAAQ,C;MACtB,iBA AiB,WA  
AiB,aAN,KAAM,EA AW,EAAX,C;MAE9B,cAAU,KA AV,C;QACI,MAAM,gCAAYB,eAAzB,C;WACV,qBAA  
M,KAAN,MAAgB,EA AhB,C;QACI,IAAI,mCAAW,MAAf,C;UAAuB,MAAM,+B;QAC7B,sBAAsB,K;QACtB,sB  
AAsB,K;QACtB,eAA8B,I;QAC9B,OAAO,QAAQ,MAAf,C;UACI,IAAI,iBAAM,KAAN,MAAgB,EAAPB,C;YAC  
I,IAAI,mBAAMB,mCAAW,MAAIc,C;CAA0C,MAAM,+B;YACHd,kBA AkB,I;YACIB,Q;WAEkB,iBA Ae,K;UA+  
EjD,QAHGc,U;UAIhC,Y;YAAO,eAhFqB,KAgFjB,O;YAAJ,S;cAAc,SAAU,YAhFH,KAgFG,YAAK,CAAL,E;cA  
AV,OAhFqC,CAAM,kBA AK,EAAL,CAAN,qCA AkB,2C;;;YAgFnC,a;;UAhF7B,gBA AgB,KiBvIcG E,WjBmqClF,  
UiBnqCkF,EjBwqCrF,CiBxqCqF,C;UjBwlChF,IAAI,SuBrhCgC,YAAU,CvBqhC9C,C;YAAyB,MAAM,+B;UAC/  
B,gBAAS,SAAU,OAA nB,I;UACqB,cAAU,K;UsBzrCpC,U;UAAA,IAAI,WAAS,CAAT,IAAc,WAAS,iBtByrCP,K  
sBzrCO,CAA3B,C;YAAA,StByrCoB,KsBzrCkB,YAAI,OAAJ,C;;YtByrCO,MAAM,gCAAYB,qCAAzB,C;;UAA9

C,qB;UACA,qB;UACA,WAAW,sBAAsB,QAAtB,EAAgC,eAAhC,C;UACX,IAAI,YAAY,IAAZ,IAAoB,yBAAY,I  
AAZ,MAAxB,C;YAA0C,MAAM,gCAAYB,yCAAzB,C;UACbD,WAAW,I;UACX,eAAyB,WAAV,SAAU,EAAQ,  
EAAR,C;UACzB,IAAI,+CAAgC,WAAW,CAA/C,C;YACI,YAAY,SiBjmCgE,WjBimC5C,CiBjmC4C,EjBimCzC,  
QiBjmCyC,C;YjBkmC5E,4BAA2C,aAAjC,0BAA0B,KAA1B,CAAiC,EAAW,IAAX,CAA3C,C;YACA,4BAAMd,  
aAAX,SAA9B,SiBtmCmD,WjBsmC/B,QiBtmC+B,CjBsmCrB,CAAW,EAAW,IAAX,CAAnD,C;;YAEA,4BAA+C  
,aAArC,0BAA0B,SAA1B,CAAqC,EAAW,IAAX,CAA/C,C;;;aAIZ,c;QACI,MAAM,+B;;QACV,IAAM,cAAN,KA  
AM,EAAc,KAAc,EAAqB,cAArB,EAAqC,CAArC,EQ/xCH,MAAO,KR+xCmD,SAAS,KAAT,IQ/xCnD,ER+xCm  
E,cAAe,OQ/xCIF,CR+xCJ,EAA4G,IAA5G,CAAN,C;UACI,SAAS,gCAAS,S;;UAIIB,iBAA8B,I;UAC9B,iBAAiB,  
K;UACjB,kBAAkB,CAAC,O;UACnB,IAAI,WAAW,iBAAM,KAAN,MAAgB,EAA3B,IAAwC,QAAN,KAAM,C  
AAN,KAAgB,EAAtD,C;YACI,cAAc,I;YACd,IAAI,oCAAW,uBAAX,EAAW,MAAX,CAAJ,C;cAAyB,MAAM,g  
CAAYB,eAAzB,C;WAEnc,OAAO,QAAQ,MAAf,C;YACI,IAAI,cAAc,WAAIB,C;cA8CZ,UA7CwC,K;cA8CxC,Y  
;gBAAO,mBA9CiB,KA8Cb,O;gBAAJ,W;kBAAc,SA9C4B,UA8CIB,YA9CP,KA8CO,YAAK,GAAL,EA9CkB,M  
AAM,E;;;gBA8Cd,iB;;cA9CzB,QA+CT,G;aA7CK,aAAa,I;YACS,mBAAe,K;YA0CjD,UAHgC,Y;YAIhC,Y;cAA  
O,mBA3CqB,KA2CjB,O;cAAJ,W;gBAAc,WAAU,YA3CH,KA2CG,YAAK,GAAL,E;gBAAV,SA3CqC,CAAM,k  
BAAK,EAAL,CAAN,uCAAkB,oBAAM,E;;;cA2CzC,iB;;YA3C7B,kBAAGB,KiB5nCgE,WjBmqClF,YiBnqCkF,Ej  
BwqCrF,GiBxqCqF,C;YjB6nChF,IAAI,WuB1jCgC,YAAU,CvB0jC9C,C;cAAyB,MAAM,+B;YAC/B,gBAAS,WA  
AU,OAAAnB,I;YACqB,mBAAe,K;YAUChD,UAHgC,Y;YAIhC,Y;cAAO,mBAxCoB,KAwChB,O;cAAJ,W;gBAAc,  
WAAU,YAxCJ,KAwCI,YAAK,GAAL,E;gBAAV,SAxCoC,CAAM,kBAAK,GAAL,CAAN,mC;;;cAwChB,iB;;YA  
xC7B,eAAe,KiB/nCiE,WjBmqClF,YiBnqCkF,EjBwqCrF,GiBxqCqF,C;YjBgoChF,gBAAS,QAAS,OAAIB,I;YAC  
A,aAAW,wBAAwB,QAAXB,C;YACX,IAAI,cAAyB,IAAZ,IAAoB,2BAAyB,MAAZ,MAAxB,C;cAA0C,MAAM,gC  
AAyB,yCAAzB,C;YACbD,aAAW,M;YACX,iBAAyB,WAAV,WAAU,EAAQ,EAAR,C;YACzB,IAAI,aAAW,CA  
Af,C;cACI,cAAyB,WiBtoCgE,WjBsoC5C,CiBtoC4C,EjBsoCzC,UiBtoCyC,C;cjBuoC5E,4BAAyB,aAAT,OAAO,  
AAM,CAAS,EAAW,MAAX,CAAzB,C;cACA,4BAAMd,aAAX,SAA9B,WiB3oCmD,WjB2oC/B,UiB3oC+B,CjB  
2oCrB,CAAW,EAAW,MAAX,CAAnD,C;cACA,IAAI,QAAQ,MAAZ,C;gBAAoB,MAAM,gCAAYB,mCAAzB,C;;  
cAE1B,4BAA6B,aAAT,OAAV,WAAU,CAAS,EAAW,MAAX,CAA7B,C;;;MAKhB,OAAW,UAAJ,GAAiB,MA  
AD,aAAhB,GAA6B,M;K;IAIxC,0C;MACI,aAAa,KAAM,O;MACnB,iBAAiB,C;MACjB,IAAI,SAAS,CAAT,IAA  
c,YAAY,IAAZ,mBAAM,CAAN,EAAIB,C;QAAoC,+B;OACbC,YAAC,SAAS,UAAT,IAAD,IAAwB,E;MAAxB,S  
;QAA4D,gBAA7B,yBAAkB,iBAAN,KAAM,CAAIB,C;QAA6B,c;;UU4ThD,U;UADhB,IAAI,wCAAsB,mBAA1B  
,C;YAAqC,aAAO,I;YAAP,e;WACrB,6B;UAAhB,OAAgB,gBAAhB,C;YAAgB,2B;YAAM,IAAI,CV5T4C,CAAa,  
kBAAK,EAAL,CAAb,oCU4TjC,OV5TiC,EU4ThD,C;cAAyB,aAAO,K;cAAP,e;;UAC/C,aAAO,I;;;QV7TyD,iB;O  
AAhE,S;QAEI,OAAW,iBAAM,CAAN,MAAY,EAAhB,sD;OAGX,OAAiB,WAAN,KAAM,EAAW,GAAX,CAAV  
,GAAyC,OAAR,QAAN,KAAM,EAAC,CAAL,CAAQ,CAAzC,GAA6D,OAAO,KAAM,C;K;IAKxE,0D;MAII,QA  
HgC,U;MAIhC,OAAO,IAAI,gBAAJ,IAJqC,SAIvB,CAAU,iCAAK,CAAL,EAAV,CAArB,C;QAAyC,a;;MAJzC,O  
iBnqC4F,oBjBmqClF,UiBnqCkF,EjBwqCrF,CiBxqCqF,C;K;IjBqqChG,qD;MACI,QAAQ,U;MACR,OAAO,IAAI,  
gBAAJ,IAAc,UAAU,iCAAK,CAAL,EAAV,CAArB,C;QAAyC,a;;MACzC,OAAO,C;K;;;IAmBX,8B;MAA+C,q  
CAAQ,OAAR,E;K;IAC/C,+B;MAAGD,2CAAS,OAAT,E;K;IAEhD,sC;MAAiD,oBAAS,sBAAGB,CAAhB,CAAT,  
C;K;IACjD,wC;MAAMd,oBAAU,uBAAiB,CAAjB,CAAD,yBAAuB,CAAvB,EAAT,C;K;IACnD,oD;MAAoE,oB  
AAU,sBAAgB,CAAhB,CAAD,yBAAsB,iBAATB,EAAT,C;K;IACpE,0C;MACI,IAAI,sEAAqB,SAArB,CAAJ,C;Q  
AAA,OACI,gBAAgB,KAAhB,C;;QADJ,OAGI,iBAAiB,cAAc,KAAc,CAAjB,C;;K;IAGR,4C;MACI,IAAI,kEAAg  
C,mBAAhC,CAAJ,C;QAAA,OACI,gBAAgB,cAAc,MAAd,CAAhB,C;;QADJ,OAGI,iBAAwB,WAAp,MAAO,yB  
AAsB,UAAAtB,CAAxB,C;;K;IuMI3CR,8B;MAEGD,QAAM,SAAN,M;aAC5C,a;UAD4C,OACbB,I;aAC5B,c;UAF4  
C,OAef,I;aAC7B,c;UAH4C,OAGf,I;aAC7B,S;UAJ4C,OAIpB,G;aACxB,S;UAL4C,OAKpB,G;aACxB,O;UAN4C,  
OAMtB,G;aActB,M;UAP4C,OAovB,G;gBnMuEwB,MAAM,6BAA8B,CmMtEnE,mBAAgB,SnMsEmD,YAA9B,  
C;;K;ImMnevD,4C;MACwE,QAAM,SAAN,C;aACpE,I;UADoE,6C;aAEpE,I;UAFoE,8C;aAGpE,I;UAHoE,8C;aA  
IpE,G;UAJoE,yC;aAKpE,G;UALoE,yC;aAMPe,G;UANoE,uC;aAOpe,G;UAPoE,sC;gBAQ5D,MAAM,gCAAYB,  
uCAAoC,SAA7D,C;;K;IAGIB,yD;MAGQ,KAAC,eAAD,C;QAEQ,IADE,OACF,Q;UAHZ,sC;;UAIoB,MAAM,gC  
AAyB,4EAAqD,OAARd,CAAzB,C;;QAIIB,QAAM,OAAO,C;eACI,E;YATZ,uC;eAUyE,YAVZ,yC;eAWyE,YA  
XZ,yC;kBAYoB,MAAM,gCAAYB,yDAaKc,OAAIC,CAAzB,C;;K;IC5F9B,4B;K;;MC4BI,kC;;IAXA,gC;MAAA

,oC;MAM0B,2BAAc,iC;K;8CACpC,Y;MAAkC,OAAA,iCAAoB,W;K;6CADhC,Y;MAAA,yC;K;;;IAN1B,4C;MAAA,2C;QAAA,0B;OAAA,oC;K;IAWA,gC;MAAA,oC;K;;;IAAA,4C;MAAA,2C;QAAA,0B;OAAA,oC;K;;;IAKJ,oB;K;qCAcI,oB;MAK8D,4BAAiB,IAAjB,EAAuB,QAAvB,C;K;sCAE9D,oB;MAK+D,wBAAM,QAAD,aAAL,C;K;sCAG/D,Y;MAMqC,QAAC,iBAAa,a;K;yCAEnD,Y;MAMwC,OAAA,iBAAa,a;K;4EAIzD,yB;MAAA,4C;MAAA,mC;QAQuE,MAAM,WAAM,0BAAN,C;O;KAR7E,C;mFAUA,yB;MAAA,4C;MAAA,mC;QAQsE,MAAM,WAAM,0BAAN,C;O;KAR5E,C;IAY8B,4C;MAAiD,mB;MAAhD,gB;MAAoB,4B;K;4CAC/C,Y;MAAsC,OAAA,SAAK,aAAL,cAAoB,eAApB,C;K;6CAEtC,oB;MAAkD,4BAAiB,SAAjB,EAAuB,4BAAa,QAAb,CAAvB,C;K;;IChGV,sC;MAAC,gB;K;IAOf,4E;MAA8G,mB;MAA7G,4B;MAA6B,8B;MAAgD,sB;K;+DACpG,Y;MAAsC,OAAgC,aAA/B,iBAAW,OAAx,UAAoB,gBAApB,CAA+B,EAAW,iBAAW,KAAtB,CAAhC,cAA8D,aAA9D,C;K;gEAcTc,oB;MAAkD,+CAAA,gBAAb,EAAwB,iBAxB,EAAoC,0BAAS,QAAT,CAApC,C;K;;+CAGtD,Y;MAAmC,+CAAA,WAAb,EAAqB,IAArB,EAA2B,gCAAS,KAAPC,C;K;;IAUO,wC;MAAC,gB;K;IAOf,gF;MAAkH,mB;MAAjH,4B;MAA+B,8B;MAAkD,sB;K;mEAC1G,Y;MAAsC,OAAgC,aAA/B,iBAAW,OAAx,GAAoB,gBAAW,EAAW,iBAAW,KAAtB,CAAhC,cAA8D,aAA9D,C;K;oEAcTc,oB;MAAkD,mDAAe,gBAAf,EAA0B,iBAA1B,EAA5C,0BAAS,QAAT,CAAtC,C;K;;iDAGtD,Y;MAAmC,mDAAe,WAAf,EAAuB,IAAvB,EAA6B,gCAAS,KAAtC,C;K;;IAGvC,0B;MAGb8B,yE;MAC1B,mB;K;oCAEA,Y;MAA4B,qB;K;iDAE5B,oB;MAWc,Q;MADV,gBAAgB,QAAS,gBAAO,SAAP,C;MACf,IAAI,gDAA+B,4CAAnC,C;QAEN,iBAAiB,mBAAU,SAAV,C;QACjB,IAAI,mBAAy,SAAZ,gBAAyB,CAAzB,IAA8B,mBAAy,UAAZ,eAAyB,CAA3D,C;UAA8D,gBAAS,QAAT,C;QAC9D,iB;;QAEA,YAAAY,QAAS,kBAAS,SAAT,C;QAErB,mBAAiB,4BAAU,K;QAC3B,IAAI,sDAA+B,kDAAAnC,C;UAAgE,gBAAS,QAAT,C;QACrD,8BAAX,YAAW,C;;MAVf,qB;K;0CAcJ,oB;MACI,MAAM,6BAASB,iDAA+C,cAA/C,qCAA0E,QAAlE,MAAtB,C;K;;qFC7Fd,yB;MAAA,yC;MAAA,wB;QA2BI,WAAW,8B;QAhB6B,KAIbxC,E;QAJBA,OakBO,IAAK,a;O;KA7BhB,C;uFAeA,4B;MAYI,WAAW,mB;MACX,O;MACA,OAAO,IAAK,a;K;IAYe,qC;MAAC,kB;MAAc,wB;K;;sCAR9C,Y;MAQgC,iB;K;sCARhC,Y;MAQ8C,oB;K;wCAR9C,2B;MAAA,sBAQgC,qCARhC,EAQ8C,8CAR9C,C;K;oCAAA,Y;MAAA,OAQgC,iDARhC,IAQ8C,8CAR9C,O;K;oCAAA,Y;MAAA,c;MAQgC,sD;MAAc,yD;MAR9C,a;K;kCAAA,iB;MAAA,4IAQgC,sCARhC,IAQ8C,4CAR9C,I;K;iGAUA,yB;MAAA,yC;MAGBA,8C;MAhBA,wB;QA6BI,WAAW,8B;QACX,aAjB8C,KAIbJc,E;QAJBb,OakBO,oBAAW,MAAX,EAAMB,IAAK,aAxB,C;O;KA/BX,C;mGAgBA,yB;MAAA,8C;MAAA,mC;QAaI,WAAW,mB;QACX,aAAa,O;QACb,OAAO,oBAAW,MAAX,EAAMB,IAAK,aAxB,C;O;KafX,C;IxJZA,2E;MASI,sC;MAAA,4C;K;IATJ,mGAWY,Y;MAAQ,2B;KAXpB,E;IAAA,4DAaQ,kB;MACI,wBAAW,MAAX,C;K;IadZ,wF;IyJewC,sC;MACpC,0B;K;;IAGJ,kC;MAUI,OAA2C,CAA3C,2BAA6B,uBAA7B,EAAoC,KAAPC,CAA2C,e;K;IAE/C,8B;K;kDAuBI,4B;MASI,MAAM,qCA8B,8CAA9B,C;K;;;IAa4B,8C;MAGtC,6B;MAEmD,UAMX,M;MAPxC,kBACmD,mE;MAEnD,eAC0B,K;MAE1B,cACwC,kE;MAExC,gBACmC,gB;K;iGAG/B,Y;MAAQ,0C;K;0DAEZ,kB;MACI,cAAY,I;MACZ,gBAAc,M;K;IAGsE,iG;MAAA,uB;QAExE,Q;QAAZ,qCAAY,8D;QACZ,sCAAA,a;QAFb,OAGA,yB;O;K;2DAJJ,+B;MAAKD,OAA5C,wDAAtC,c;K;IAOyE,uH;MAAA,uB;QAExG,Q;QAaf,iBAAe,8F;QACf,eAAK,2B;QAA6B,mC;QrMjGtB,gBAAT,Q;QqMsG0D,kB;QAJzD,sBAASB,SAAK,W;QAC3B,IAAI,eAAa,eAAjB,C;UAEL,iC;UACA,mBAAy,oCAAwB,eAAXB,EAAyC,kEAAzC,C;;UAGZ,mBAAy,kE;;QAEhB,oBAAa,e;QAZjB,OAcA,yB;O;K;6DAfJ,0C;MAAqF,OAA5C,qEAAAtC,c;K;IAqBzB,mI;MAAA,qB;QACxD,yCAAgB,uB;QAGhB,qCAAY,Y;QACZ,uCAAc,E;QACIB,W;O;K;IEATA,iC;MAGwB,wCAAA,mCAAb,EAAoC,kFAAPC,C;K;mDAQxB,Y;MAMuB,UADC,MACD,EAIH,MAJG,EAaK,M;MAjBxB,OAAO,IAAP,C;QAEI,aAAa,IAAK,S;QACF,SAAL,IAAK,O;QAAL,mB;UACyB,gBAArB,0D;U1JxBhB,U;UADP,yB;U0JyBe,O1JxBR,sF;S0JuBC,WAAW,M;QAGX,IAAI,mDAAoB,MAApB,QAAl,C;;YAlIb,SAAT,exJxJV,CwJwJuD,IxJxJvD,EwJwJ6D,YxJxJ7D,EwJwJoE,IxJxJpE,EAA8C,KAA9C,C;;YwJyJQ,gC;cACE,IzJzJhB,oBDgDQ,WAAO,c0JyG0B,C1JzG1B,CAAP,CChDR,C;cyJ0JgB,Q;;cALI,O;;UAAR,c;UAQA,IAAI,MAAM,yBAAV,C;YACI,IzJvKhB,oBDgDQ,W0JuHoB,0E1JvHpB,CChDR,C;;UyJ0KY,gBAAc,gB;UACd,IAAK,oBAAW,MAAX,C;;;K;;0EC1MrB,4B;MAoKI,QAhKK,SAGKG,GAhKoB,KAgKpB,I;MACR,IAAI,CAjKC,SAiKD,GAjKwB,KAIkxB,IAAiB,CAAjB,IAAsB,eAjKE,KAIkF,MAjKrB,SAiKL,C;QAA6C,a;OAJK7C,OakKO,C;K;kEAhKX,yB;MAAA,0B;MAAA,mC;QA2KI,QAnKK,SAmKG,GAnKe,K;QAAvB,OAAgC,OAOkzB,KApKgB,KAOkX,GAAW,CAAC,CAAC,IAPkF,KAOkC,KAAmB,KAAC,CAAC,CAAD,IAAL,CAAnB,CAAD,KAAC,EAAID,KApKyB,C;O;KARpC,C;4EAUA,4B;MAoJI,QAhJK,SAGJG,GAhJoB,KAgJpB,I;MACR,IAAI,CAjJC,SAiJD,GAjJwB,KAIjxB,IAAiB,CAAjB,IAAsB,eAjJE,KAIjF,MAjJrB,SAiJL,C;QAA6C,a;OAJJ7C,OakJO,C;k;KEAhJ

X,yB;MAAA,4B;MAAA,mC;QA2JI,QAnJK,SAmJG,GAnJe,K;QAAvB,OAAgC,QAoJzB,KApJgB,KAoJX,GAAW,CAAC,CAAC,IAPJF,KAoJC,KAAmB,KAAK,CAAC,CAAD,IAAL,CAAnB,CAAD,KAAkC,EAAID,KApJyB,C;O;KARpC,C;4EAUA,4B;MAoII,QAhIK,SAGIG,GAhIc,KAgId,I;MACR,IAAI,CAjIC,SAiID,GAjIkB,KAiIB,IAAiB,CAAjB,IAAsB,eAjIJ,KaiII,MAjIrB,SAiIL,C;QAA6C,a;OAJI7C,OAKIO,C;K;KEAhIX,4B;MA2II,QAnIK,SAmIG,GAnIS,K;MAAjB,OAoIO,KApIU,KAoIL,GAAW,CAAC,CAAC,IAPIR,KAoIO,KAAmB,KAAK,CAAC,CAAD,IAAL,CAAnB,CAAD,KAAkC,EAAID,K;K;4EAIIX,yB;MAqMA,0B;MArMA,mC;QAikB,kBAAT,oBAAL,SAAK,C;QAqML,QAAQ,gBArMe,KaqMf,C;QACR,IAAI,gBAtmM,KAsMnB,eAAiB,CAAjB,IAAsB,mBAtmH,KAsMG,GAAa,WAAb,CAA1B,C;UAA6C,W;SatM7C,OAuMO,C;O;KA3MX,C;KEAMA,4B;MAGNI,QAxMK,oBAAL,SAAK,CAwMG,QAxMU,KAwMV,C;MAxMR,OAYMO,MAzMW,KAYMN,KAAa,MAzMP,KAYMO,CAAD,KAAmB,KAAM,CAAD,aAAL,CAAnB,CAAD,YAAkC,EAAIC,CAAX,CAAL,C;K;4EAvmX,4B;MAoGI,QAhhGK,SAGGG,GAhGoB,KAgGpB,I;MACR,IAAI,CAjGC,SAiGD,GAjGwB,KAiGxB,IAAiB,CAAjB,IAAsB,eAjGE,KAiGF,MAjGrB,SAiGL,C;QAA6C,a;OAJG7C,OAKGO,C;K;KEAhGX,yB;MAAA,0B;MAAA,mC;QA2GI,QAnGK,SAmGG,GAnGe,K;QAAvB,OAAgC,OAoGzB,KApGgB,KAoGX,GAAW,CAAC,CAAC,IAPGF,KAoGC,KAAmB,KAAK,CAAC,CAAD,IAAL,CAAnB,CAAD,KAAkC,EAAID,KApGyB,C;O;KARpC,C;4EAUA,4B;MAoFI,QAhhFK,SAGFG,GAhFoB,KAgFpB,I;MACR,IAAI,CAjFC,SAiFD,GAjFwB,KAiFxB,IAAiB,CAAjB,IAAsB,eAjFE,KAiFF,MAjFrB,SAiFL,C;QAA6C,a;OAJF7C,OAKFO,C;K;KEAhFX,yB;MAAA,4B;MAAA,mC;QA2FI,QAnFK,SAmFG,GAnFe,K;QAAvB,OAAgC,QAoFzB,KApFgB,KAoFX,GAAW,CAAC,CAAC,IAPFF,KAoFC,KAAmB,KAAK,CAAC,CAAD,IAAL,CAAnB,CAAD,KAAkC,EAAID,KApFyB,C;O;KARpC,C;4EAUA,4B;MAoEI,QAhhEK,SAGEG,GAhEc,KAgEd,I;MACR,IAAI,CAjEC,SAiED,GAjEkB,KAiEIB,IAAiB,CAAjB,IAAsB,eAjEJ,KAiEI,MAjErB,SAiEL,C;QAA6C,a;OAJE7C,OAKEO,C;K;KEAhEX,4B;MA2EI,QAnEK,SAmEG,GAnES,K;MAAjB,OAoEO,KApEU,KAoEL,GAAW,CAAC,CAAC,IAPER,KAoEO,KAAmB,KAAK,CAAC,CAAD,IAAL,CAAnB,CAAD,KAAkC,EAAID,K;K;4EAIEX,yB;MAqIA,0B;MArIA,mC;QAikB,kBAAT,oBAAL,SAAK,C;QAqIL,QAAQ,gBArLe,KaqIf,C;QACR,IAAI,gBAtmB,KAsInB,eAAiB,CAAjB,IAAsB,mBAtmH,KAsIG,GAAa,WAAb,CAA1B,C;UAA6C,W;SatI7C,OAuIO,C;O;KA3IX,C;KEAMA,4B;MAGJI,QAxIK,oBAAL,SAAK,CAwIG,QAxIU,KAwIV,C;MAxIR,OAYIO,MAzIW,KAYIN,KAAa,MAZIP,KAYIO,CAAD,KAAmB,KAAM,CAAD,aAAL,CAAnB,CAAD,YAAkC,EAAIC,CAAX,CAAL,C;K;2EAvmX,4B;MAoCI,QAhCA,SAGCQ,GAhCY,KAgCZ,I;MACR,IAAI,CAjCJ,SAiCI,GAjCgB,KAiChB,IAAiB,CAAjB,IAAsB,eAjCN,KAiCM,MAjC1B,SAiCA,C;QAA6C,a;OAJC7C,OAKCO,C;K;KEAhCX,yB;MAAA,0B;MAAA,mC;QA2CI,QAnCA,SAmCQ,GAnCO,K;QAaf,OAAwB,OAoCjB,KApCQ,KAoCH,GAAW,CAAC,CAAC,IAPCV,KAoCS,KAAmB,KAAK,CAAC,CAAD,IAAL,CAAnB,CAAD,KAAkC,EAAID,KApCiB,C;O;KAR5B,C;4EAUA,4B;MAoBI,QAhBA,SAGBQ,GAhBY,KAgBZ,I;MACR,IAAI,CAjBJ,SAiBI,GAjBgB,KAiBhB,IAAiB,CAAjB,IAAsB,eAjBN,KAiBM,MAjB1B,SAiBA,C;QAA6C,a;OAJB7C,OAKBO,C;K;MEAhBX,yB;MAAA,4B;MAAA,mC;QA2BI,QAnBA,SAmBQ,GAnBO,K;QAaf,OAAwB,QAoBjB,KApBQ,KAoBH,GAAW,CAAC,CAAC,IAPBV,KAoBS,KAAmB,KAAK,CAAC,CAAD,IAAL,CAAnB,CAAD,KAAkC,EAAID,KApBiB,C;O;KAR5B,C;4EAUA,4B;MAII,QAAQ,YAAO,KAAP,I;MACR,IAAI,aAAS,KAAI,IAAiB,CAAjB,IAAsB,eAAI,KAAAJ,MAAA,SAAvC,C;QAA6C,a;OAC7C,OAAO,C;K;MEAGX,4B;MAQI,QAAQ,YAAO,K;MACf,OAAO,KAAK,QAAW,CAAC,CAAC,IAAM,KAAP,KAAmB,KAAK,CAAC,CAAD,IAAL,CAAnB,CAAD,KAAkC,EAAID,K;K;4EAGX,yB;MAGEA,0B;MAhEA,mC;QAikB,kBAAT,oBAAL,SAAK,C;QAgEL,QAAQ,gBAhEe,KAgEf,C;QACR,IAAI,gBAjEmB,KAiEnB,eAAiB,CAAjB,IAAsB,mBAjEH,KAiEG,GAAa,WAAb,CAA1B,C;UAA6C,W;SAjE7C,OAKEO,C;O;KATeX,C;KEAMA,4B;MA2EI,QAnEK,oBAAL,SAAK,CAMEG,QAnEU,KAmEV,C;MAnER,OAoEO,MApEW,KAoEN,KAAa,MApEP,KAoEO,CAAD,KAAmB,KAAM,CAAD,aAAL,CAAnB,CAAD,YAAkC,EAAIC,CAAX,CAAL,C;K;6EAIEX,yB;MAGDA,0B;MAhDA,mC;QAIS,cAAe,oBAAN,KAAm,C;QAgDpB,QAhDA,SAGDQ,KAAO,OAAP,C;QACR,IAjDA,SAiDI,KAAO,OAAT,eAAiB,CAAjB,IAAsB,mBAAI,OA AJ,GAjD1B,SAiD0B,CAA1B,C;UAA6C,W;SAjD7C,OAKDO,C;O;KATDX,C;MEAMA,yB;MAAA,0B;MAAA,mC;QAQS,cAAU,oBAAN,KAAm,C;QAmDf,QAnDA,SAmDQ,QAAO,OAAP,C;QAnDR,OAAYB,OAoDIB,MAAK,YAAa,MAAM,OAAN,CAAD,KAAmB,KAAM,CAAD,aAAL,CAAnB,CAAD,YAAkC,EAAIC,CAAX,CAAL,CAPDkB,S;O;KAR7B,C;6EAUA,yB;MAGCA,0B;MAhCA,mC;QAIS,cAAe,oBAAN,KAAm,C;QAgCpB,QAhCA,SAGCQ,KAAO,OAAP,C;QACR,IAjCA,SAiCI,KAAO,OAAT,eAAiB,CAAjB,IAAsB,mBAAI,OA AJ,GAjC1B,SAiC0B,CAA1B,C;UAA6C,W;SAjC7C,OAKCO,C;O;KATCX,C;MEAMA,yB;MAAA,4B;MAAA,mC;QAQS,cAAU,oBAAN,KAAm,C;QAmC

f,QAnCA,SAmCQ,QAAO,OAAP,C;QAnCR,OAAYB,QAoCIB,MAAK,YAAa,MAAM,OAAN,CAAD,KAAmB,K  
AAM,CAAD,aAAL,CAAnB,CAAD,YAAkC,EAAIC,CAAX,CAAL,CAPcKb,S;O;KAR7B,C;6EUA,yB;MAGBA  
,0B;MAhBA,mC;QAIS,cAAe,oBAAN,KAAM,C;QAgBpB,QAhBA,SAGBQ,KAAO,OAAP,C;QACR,IAjBA,SaiB  
I,KAAS,OAAT,eAAiB,CAAjB,IAAsB,mBAAI,OAjB,GAjB1B,SaiB0B,CAA1B,C;UAA6C,W;SAjB7C,OakBO,  
C;O;KAtBX,C;mEAMA,4B;MAQS,cAAU,oBAAN,KAAM,C;MAMbF,QAnBA,SAmBQ,QAAO,OAAP,C;MAnB  
R,OAoBO,MAAK,YAAa,MAAM,OAAN,CAAD,KAAmB,KAAM,CAAD,aAAL,CAAnB,CAAD,YAAkC,EAAIC  
,CAAX,CAAL,CAPcKb,Q;K;6EAE7B,yB;MAAA,0B;MAAA,mC;QAI,QAAQ,cAAO,KAAP,C;QACR,IAAI,cA  
AS,KAAT,eAAiB,CAAjB,IAAsB,mBAAI,KAJ,GAAa,SAAb,CAA1B,C;UAA6C,W;SAC7C,OAAO,C;O;KANX,  
C;mEASA,4B;MAQI,QAAQ,iBAAO,KAAP,C;MACR,OAAO,MAAK,UAAa,MAAM,KAAN,CAAD,KAAmB,K  
AAM,CAAD,aAAL,CAAnB,CAAD,YAAkC,EAAIC,CAAX,CAAL,C;K;kEAGX,yB;MpGiqB2C,iB;MoGjqB3C,m  
C;QAU,QAAQ,YAAO,K;QACJ,iBAAS,G;QAAT,S;UAAsB,OpGspBc,MAAiC,MoGtpB/C,CpGspB+C,CoGtpB/  
C,KpGspBc,MAAiC,MoGtpBrC,KpGspBqC,C;SoGtpBhF,OAAO,OAAGD,IAAI,KAAPD,GAA+D,C;O;KAX1E,C;  
mEAcA,yB;MpG0I6C,iB;MoG1I7C,mC;QAKCI,QAxBK,SAwBG,GAXBY,K;QAYBT,iBAAK,G;QAAL,S;UAYY,  
OpGuG0B,MAAW,MoGvGrC,CpGuGqC,CoGvGrC,KpGuG0B,MAAW,MoGhIxC,KpGgIwC,C;SoGhI5D,OAYB  
O,OAAsC,IAzBzB,KAYBb,GAAqD,C;O;KAnChE,C;mEAYA,yB;MpG8H6C,iB;MoG9H7C,mC;QASBI,QAZA,S  
AYQ,GAZO,K;QAaJ,iBAAK,G;QAAL,S;UAYY,OpGuG0B,MAAW,MoGvGrC,CpGuGqC,CoGvGrC,KpGuG0B,  
MAAW,MoGpH7C,KpGoH6C,C;SoGpH5D,OAaO,OAAsC,IAb9B,KAaR,GAAqD,C;O;KAvBhE,C;mEAYA,yB;  
MpGkH6C,iB;MoGIH7C,mC;QAU,QAAQ,YAAO,K;QACJ,iBAAK,G;QAAL,S;UAYY,OpGuG0B,MAAW,MoG  
vGrC,CpGuGqC,CoGvGrC,KpGuG0B,MAAW,MoGvG3B,KpGuG2B,C;SoGvG5D,OAAO,OAAsC,IAAI,KAA1C,  
GAAqD,C;O;KAXhE,C;4ECnTA,yB;MAAA,8B;MAAA,4B;QAOyC,Q;QAAA,gFAAoB,C;O;KAP7D,C;ICM0B,4  
C;MA+CtB,qC;MA/CuB,kB;MAAgB,kB;MAAgB,kB;MAMvD,iBAAsB,iBAAU,UAAV,EAAiB,UAAjB,EAAwB,  
UAAxB,C;K;OCAEtB,+B;M3MWA,IAAI,E2MViB,CAAT,sBAAY,GAAZ,KAA4C,CAAT,sBAAY,GAA/C,MAA  
+E,CAAT,sBAAY,GAAIF,C3MUR,CAAJ,C;QACI,c2MVI,2E;Q3MWJ,MAAM,gCAAyB,OAAQ,WAAjC,C;O2M  
TN,OAAO,CAAA,KAAM,IAAI,EAAV,KAAgB,KAAM,IAAI,CAA1B,IAA+B,KAA/B,I;K;uCAGX,Y;MAGkC,O  
AAE,UAAF,oBAAS,UAAT,SAAGB,U;K;qCAEID,iB;MAEWB,gB;MADpB,IAAI,SAAS,KAAb,C;QAAoB,OAAO  
,I;MACP,iE;MAAD,mB;QAA6B,OAAO,K;OAAvD,mBAAmB,M;MACnB,OAAO,IAAK,UAAAL,KAAgB,YAAa,  
U;K;uCAGxC,Y;MAA+B,qB;K;8CAE/B,iB;MAAoD,wBAAU,KAAM,UAAhB,I;K;gDAEpD,wB;MAKI,OAAA,I  
AAK,MAAL,GAAa,KAAb,KAAuB,IAAK,MAAL,KAAc,KAAd,IACf,IAAK,MAAL,IAAc,KADtB,C;K;gDAGJ,+  
B;MAKI,OAAA,IAAK,MAAL,GAAa,KAAb,KAAuB,IAAK,MAAL,KAAc,KAAd,KACd,IAAK,MAAL,GAAa,K  
AAb,KAAsB,IAAK,MAAL,KAAc,KAAd,IACf,IAAK,MAAL,IAAc,KADrB,CADc,CAAvB,C;K;IAIJ,mC;MAAA  
,uC;MACI,2BAIuC,G;MAEvC,eAIoC,uCAA0B,M;K;IAXIE,+C;MAAA,8C;QAAA,6B;OAAA,uC;K;IA9CA,iD;  
MAAA,uD;MAG6C,0BAAK,KAAL,EAAy,KAAZ,EAAmB,CAAnB,C;MAH7C,Y;K;IA6DJ,qC;MAAA,yC;K;8C  
AEI,Y;MAC2B,yBAAC,CAAd,EAAiB,CAAjB,EAAoB,EAApB,C;K;IAH/B,iD;MAAA,gD;QAAA,+B;OAAA,yC  
;K;4FCxDI,yB;MAAA,2D;MAAA,4B;QAAQ,MAAM,6BAAoB,6BAApB,C;O;KAAAd,C;ICSJ,uB;MAG2C,+BA  
AoB,KAApB,C;K;4EAE3C,wC;MAO4F,sB;K;IAE5F,6C;MAAA,e;MAAA,iB;MAAA,uB;K;IAAA,2C;MAAA,8C  
;O;MAKI,wF;MAKA,sF;MAMA,wE;K;IAXA,yD;MAAA,iC;MAAA,iD;K;IAKA,wD;MAAA,iC;MAAA,gD;K;I  
AMA,iD;MAAA,iC;MAAA,yC;K;IAhBJ,uC;MAAA,iJ;K;IAAA,4C;MAAA,a;aAAA,c;UAAA,sD;aAAA,a;UAA  
A,qD;aAAA,M;UAAA,8C;gBAAA,gE;K;IAyBA,+B;MAAA,mC;K;IAAA,2C;MAAA,0C;QAAA,yB;OAAA,m  
C;K;IAGoC,qC;MACHc,qBAAsC,W;MACtC,gBAA2B,iC;K;uFAGvB,Y;MAMW,Q;MALP,IAAI,kBAAW,iCAAf  
,C;QACI,gBAAS,mC;QACT,qBAAC,I;OAGIB,OAAO,gF;K;6CAGf,Y;MAAwC,yBAAW,iC;K;wCAEnD,Y;MAA  
kC,OAAI,oBAAJ,GAA2B,SAAN,UAAM,CAA3B,GAA2C,iC;K;8CAE7E,Y;MAAkC,+BAAoB,UAApB,C;K;IA  
GG,oC;MAAC,4B;K;wEAAA,Y;MAAA,2B;K;kDAEtC,Y;MAAwC,W;K;6CAExC,Y;MAAkC,OAAm,SAAN,UA  
AM,C;K;oFC2C5C,yB;MAAA,gD;MAAA,4B;QAM6C,OAAmB,aAAIB,YAAy,GAAM,C;O;KANhE,C;oGAQA,  
yB;MxG7FA,iB;MwG6FA,4B;QAMqD,OxG7FM,MAAO,OwG6FZ,YAAy,GxG7FA,CwG6Fb,GAA6C,EAA7C,I;  
O;KANrD,C;sGAQA,yB;MAAA,kE;MAAA,4B;QAMsD,OAAmB,sBAAIB,YAAW,GAAO,C;O;KANzE,C;8FAQ  
A,yB;MAAA,0D;MAAA,0B;MAAA,4B;QAOMD,OAAuC,OAApB,kBAAIB,YAAy,GAAM,CAAoB,C;O;KAP1F  
,C;4FASA,yB;MAAA,wD;MAAA,0B;MAAA,4B;QAOKD,OAA2B,OAAAnB,iBAAR,SAAQ,CAAmB,C;O;KAP7E,  
C;IAUA,2C;MAaI,OAA+E,OAA9E,SAAQ,KAAI,WAAa,CAAJB,CAAR,GAaKd,CAAIB,YAAy,GAAM,MAAK,

CAAL,IAAU,WAAa,CAAvB,CAA4B,C;K;IAEnF,4C;MAaI,OAA+E,OAA9E,SAAQ,IAAI,CAAJ,IAAS,WAAa,C  
AAtB,CAAR,GAAwD,CAAIB,YAAY,GAAM,OAAK,WAAa,CAAIB,CAAsB,C;K;oFAEnF,yB;MAAA,gD;MAA  
A,4B;QAM8C,OAAqB,aAApB,YAAY,KAAQ,C;O;KANnE,C;oGAQA,yB;MxGtKA,iB;MwGsKA,4B;QAOI,OxG  
vKuD,MAAO,OwGuK7D,YAAY,KxGvKiD,CwGuK9D,GAA+C,EAA/C,I;O;KAPJ,C;sGASA,yB;MAAA,kE;MA  
AA,4B;QAMuD,OAAqB,sBAApB,YAAW,KAAS,C;O;KAN5E,C;8FAQA,yB;MAAA,0D;MAAA,4B;MAAA,4B;  
QAOqD,OAAyC,QAAPB,kBAAPB,YAAY,KAAQ,CAAOB,C;O;KAP9F,C;4FASA,yB;MAAA,wD;MAAA,4B;M  
AAA,4B;QAOoD,OAA2B,QAAnB,iBAAR,SAAQ,CAAmB,C;O;KAP/E,C;IAUA,2C;MAaI,OAAoF,QAAnF,SAA  
Q,KAAL,WAAa,EAAjB,CAAR,GAAqD,CAAPB,YAAY,KAAQ,MAAK,EAAL,IAAW,WAAa,EAAXB,CAA8B,C;  
K;IAExF,4C;MAaI,OAAoF,QAAnF,SAAQ,IAAI,EAJ,IAAU,WAAa,EAAvB,CAAR,GAA4D,CAAPB,YAAY,K  
AAQ,OAAK,WAAa,EAAlB,CAAuB,C;K;0E9MIRxY,yB;MAaA,kF;MAbA,wB;QAUbI,IAAI,CAbI,KAAR,C;UAC  
I,cAda,qB;UAeb,MAAM,8BAAYB,OAAQ,WAAjC,C;U;KAZBd,C;0EAaA,yB;MAAA,kF;MAAA,qC;QAU,IAAI,  
CAAC,KAAL,C;UACI,cAAc,a;UACd,MAAM,8BAAYB,OAAQ,WAAjC,C;U;KAZd,C;sFAGBA,yB;MAWA,kF;M  
AXA,wB;QAQW,yB;QAEp,IAfsB,KAelB,QAaj,C;UACI,cAh2B,0B;UAIb3B,MAAM,8BAAYB,OAAQ,WAAjC  
,C;UAEN,wBAnBkB,K;;QAAtB,4B;O;KARJ,C;wFAWA,yB;MAAA,kF;MAAA,qC;QAYI,IAAI,aAAJ,C;UACI,c  
AAc,a;UACd,MAAM,8BAAYB,OAAQ,WAAjC,C;UAEN,OAAO,K;;O;KAhBf,C;oEAoBA,yB;MAaA,4E;MAbA,  
wB;QAUbI,IAAI,CAbE,KAAn,C;UACI,cAdW,e;UAeX,MAAM,2BAAsB,OAAQ,WAA9B,C;U;KAZBd,C;sEAaA,  
yB;MAAA,4E;MAAA,qC;QAU,IAAI,CAAC,KAAL,C;UACI,cAAc,a;UACd,MAAM,2BAAsB,OAAQ,WAA9B,  
C;U;KAZd,C;kFAGBA,yB;MAcA,4E;MAdA,wB;QAWW,uB;QAEp,IAfoB,KAehB,QAaj,C;UACI,cAhByB,0B;U  
AiBzB,MAAM,2BAAsB,OAAQ,WAA9B,C;UAEN,sBAnBgB,K;;QAAPB,0B;O;KAXJ,C;oFAcA,yB;MAAA,4E;  
MAAA,qC;QAYI,IAAI,aAAJ,C;UACI,cAAc,a;UACd,MAAM,2BAAsB,OAAQ,WAA9B,C;UAEN,OAAO,K;;O;  
KAhBf,C;oEAqBA,yB;MAAA,4E;MAAA,0B;QAMiD,MAAM,2BAAsB,OAAQ,WAA9B,C;O;KANvD,C;I8CnHi  
C,uB;MA2D7B,8B;MA1DA,kB;K;mFAS8B,Y;MAAQ,iD;K;mFAMR,Y;MAAQ,gD;K;wFAItC,yB;MAAA,gB;M  
AAA,8B;MAAA,mB;QAWgB,Q;QADR,mB;UADJ,OACiB,I;;UADjB,OAeY,2E;O;KAXhB,C;uCAcA,Y;MAQQ,  
kBADE,UACf,kB;QADJ,OACkB,UAAM,U;;QADxB,OAeY,I;K;gCAGhB,Y;MAOQ,kBADE,UACf,kB;QADJ,O  
ACkB,UAAM,W;;QADxB,OAeY,sBAAU,UAAV,O;K;IAKhB,4B;MAAA,gC;K;wHAKI,yB;MAAA,iC;MAAA,w  
B;QAOI,uBAAO,KAAP,C;O;KAPJ,C;wHASA,yB;MAAA,kD;MAAA,iC;MAAA,4B;QAOI,uBAAO,cAAc,SAAd,  
CAAP,C;O;KAPJ,C;;IAJ,wC;MAAA,uC;QAAA,sB;OAAA,gC;K;IAwBsB,mC;MACIB,0B;K;sCAGA,iB;MAA4  
C,+CAAOB,uBAAa,KAAM,UAAnB,C;K;wCACHe,Y;MAA+B,OAAU,SAAV,cAAU,C;K;wCACzC,Y;MAAKC,o  
BAAU,cAAV,M;K;;gCA/FIC,Y;MAAA,c;MAOI,sD;MAPJ,a;K;8BAAA,iB;MAAA,2IAOI,sCAPJ,G;K;IAmGA  
,kC;MAOI,OAAO,mBAAQ,SAAR,C;K;IAEX,mC;MAQI,IAAI,8CAAJ,C;QAA6B,MAAM,eAAM,U;K;gFAG7C,  
yB;MAAA,4B;MAAA,qB;MAxqC,kD;MAwCR,wB;QAOW,Q;;UACI,OAIDH,WakDW,OAIDX,C;;UAmDN,gC;  
YACS,OA3CH,WAAO,cA2CI,CA3CJ,CAAP,C;;YAwCD,O;;QAAP,W;O;KAPJ,C;kFAcA,yB;MAAA,4B;MAAA,  
qB;MAtdQ,kD;MAAsDR,mC;QAOW,Q;;UACI,OAHEH,WAgEW,gBAhEX,C;;UAIEN,gC;YACS,OAzDH,WAAO,  
cAyDI,CAzDJ,CAAP,C;;YAsDD,O;;QAAP,W;O;KAPJ,C;8EAgBA,yB;MAAA,oD;MAAA,gB;MAAA,8B;MAAA  
,4B;QAUW,Q;QADP,yB;QACA,OAAO,gF;O;KAVX,C;+EAaA,yB;MAAA,gB;MAAA,8B;MAAA,uC;QAegB,U  
ADL,M;QAAM,gBAAGB,2B;QACzB,sB;UAAQ,yF;;UACA,mBAAU,SAAV,C;QAFZ,a;O;KADJ,C;kFAoBA,yB;  
MAAA,gB;MAAA,8B;MAAA,0C;QAUW,Q;QADP,IAAI,mBAAJ,C;UAAe,OAAO,Y;QACTB,OAAO,gF;O;KAV  
X,C;qEAaA,yB;MAAA,gB;MAAA,8B;MAAA,kD;QAIb0B,UADf,M;QAAM,gBAAGB,2B;QACzB,sB;UAAQ,m  
BAAU,gFAAV,C;;UACA,mBAAU,SAAV,C;QAFZ,a;O;KAhBJ,C;mEAwBA,yB;MAAA,4B;MAAA,gB;MAAA,8  
B;MAAA,uC;YAe8C,I;YADnC,M;QACH,wB;UAAa,gB;UAAO,SA7JhB,WA6JwB,UAAU,gFAAV,CA7JxB,C;;U  
A8JI,oBAAO,eAAP,C;QAFZ,a;O;KADJ,C;gFAoBA,yB;MAAA,gB;MAAA,8B;MAAA,iC;MA1GA,qB;MAtdQ,k  
D;MAgKR,uC;QAWW,Q;QACH,wB;UA/GG,U;;YA+GkC,U;YA9G9B,SAhEH,gBA8KuB,UAAU,sFAAV,CA9K  
vB,C;;YAIEN,gC;cACS,SAzDH,gBAAO,cAyDI,CAzDJ,CAAP,C;;cAsDD,O;;UA+GU,a;;UACL,uBAAO,eAAP,C;  
QAFZ,W;O;KAXJ,C;wEAIbA,yB;MAAA,4B;MAAA,uC;QAeW,Q;QAAM,gBAAGB,2B;QACzB,sB;UAAQ,gB;;  
UACO,OAnMX,WAmMmB,UAAU,SAAV,CAnMnB,C;;QAIrM,W;O;KADJ,C;wFAoBA,yB;MA/IA,4B;MAAA,q  
B;MAtdQ,kD;MAqMR,uC;QAWW,Q;QAAM,gBAAGB,2B;QACzB,sB;UAAQ,gB;;UApJL,U;;YACI,SAhEH,WA  
oNkB,oBApNIB,C;;YAIEN,gC;cACS,SAzDH,WAAO,cAyDI,CAzDJ,CAAP,C;;cAsDD,O;;UAqJK,a;;QAFZ,W;O;  
KAXJ,C;4EAmBA,6B;MAUI,Q;MAAA,iD;QAAYB,Y;OACzB,OAAO,S;K;4EAGX,yB;MAAA,gB;MAAA,8B;M

AAA,oC;QAU0B,Q;QAAtB,IAAI,mBAAJ,C;UAAe,OAAO,gFAAP,C;SACf,OAAO,S;O;KAXX,C;I3CtTgC,sC;M AAC,uB;QAAA,UAAkB,kC;mBAA4C,O;;K;;0DAE/F,yB;MAAA,2D;MAAA,mB;QAKoC,MAAM,8B;O;KAL1C, C;oEAOA,yB;MAAA,2D;MAAA,yB;QAMkD,MAAM,6BAAoB,sCAAmC,MAAvD,C;O;KANxD,C;gEUAU,iB; MAUI,OAAO,O;K;kEAGX,4B;MAUI,OAAO,gB;K;oEAGX,2B;MAUI,OAAgB,MAAT,QAAS,C;K;oEAGpB,4B; MAUI,gB;MACA,OAAO,S;K;kEAGX,4B;MAWI,MAAM,SAAN,C;MACA,OAAO,S;K;kEAGX,4B;MAUI,OAA O,MAAM,SAAN,C;K;sEAGX,gC;MAWI,OAAW,UAAU,SAAV,CAAJ,GAAqB,SAArB,GAA+B,I;K;8EAG1C,g C;MAWI,OAAW,CAAC,UAAU,SAAV,CAAL,GAAsB,SAAtB,GAAgC,I;K;wEAG3C,yB;MAWI,iBAAc,CAAd, UAAsB,KAAtB,U;QACI,OAAO,KAAP,C;;K;wE4MjJR,iB;MAIkF,Y;K;ICY9C,6B;MACHc,kB;MACA,oB;K;8B AGA,Y;MAGyC,aAAG,UAAH,UAAW,WAAX,M;K;;gCAvB7C,Y;MAGBI,iB;K;gCAhBJ,Y;MAiBI,kB;K;kCAjB J,yB;MAAA,gBAGBI,qCAhBJ,EAIbI,wCAjBJ,C;K;8BAAA,Y;MAAA,c;MAGBI,sD;MACA,uD;MAjBJ,a;K;4BA AA,iB;MAAA,4IAGBI,sCAhBJ,IAiBI,wCAjBJ,I;K;IA0BA,6B;MAMoD,gBAAK,SAAL,EAAW,IAAX,C;K;IAEp D,8B;MAI8C,iBAAO,eAAP,EAAc,gBAAd,E;K;IAiBD,sC;MACzC,kB;MACA,oB;MACA,kB;K;gCAGA,Y;MAG yC,aAAG,UAAH,UAAW,WAAX,UAAoB,UAApB,M;K;;kCAx7C,Y;MAGBI,iB;K;kCAhBJ,Y;MAiBI,kB;K;kC AjBJ,Y;MAkBI,iB;K;oCAIbJ,gC;MAAA,kBAGBI,qCAhBJ,EAIbI,wCAjBJ,EAKBI,qCAIbJ,C;K;gCAA,Y;MAA A,c;MAGBI,sD;MACA,uD;MACA,sD;MAIBJ,a;K;8BAAA,iB;MAAA,4IAGBI,sCAhBJ,IAiBI,wCAjBJ,IAKBI,sCA IBJ,I;K;IA2BA,8B;MAImD,iBAAO,eAAP,EAAc,gBAAd,EAAsB,eAAtB,E;K;I5NIE1B,qB;MAErB,6B;MAFwD,g B;K;IAExD,2B;MAAA,+B;MACI,iBAGoC,UAAM,CAAN,C;MAEpC,iBAGoC,UAAM,MAAN,C;MAEpC,kBAG mC,C;MAEnC,iBAGkC,C;K;;;IANtC,uC;MAAA,sC;QAAA,qB;OAAA,+B;K;kGAsBA,iB;MAOmE,OAAa,0BA2 O1C,SAAL,GAAiB,GA3O8B,EAAU,KA2OpD,KAAL,GAAiB,GA3O8B,C;K;sGAehF,iB;MAM2D,OAAa,0BAm O1C,SAAL,GAAiB,GAnOsB,EAAU,KEoO5C,KAAL,GAAiB,KFpOsB,C;K;sGAExE,yB;MA0PA,6B;MC3PA,8C; MDCA,wB;QAMyD,OCAS,YAAiB,CD6PhD,cAAU,SAAL,GAAiB,GAAtB,CC7PgD,MAAjB,EDAe,KCAc,KAA 7B,C;O;KDNIE,C;sGAQA,yB;MA4PA,WAS6D,wB;MAT7D,+B;MiB7PA,gD;MjBCA,wB;QAM0D,OiBAS,aAAk B,CjB+PhD,eAAW,oBAAL,SAAK,CAAL,UAAN,CiB/PgD,MAAIB,EjBAGB,KiBAc,KAA9B,C;O;KjBNnE,C;4F AQA,yB;MA00A,6B;MA10A,wB;QAEsD,OCMD,cAAU,CD2O5B,cAAU,SAAL,GAAiB,GAAtB,CC3O4B,MA AK,GAAW,CD2O5C,cAjPsC,KAIp5B,KAAL,GAAiB,GAAtB,CC3O4C,MAAX,IAAf,C;O;KDRrD,C;4FAGA,yB; MAuOA,6B;MAvOA,wB;QAEuD,OCGF,cAAU,CD2O5B,cAAU,SAAL,GAAiB,GAAtB,CC3O4B,MAAK,GAA W,CC4O5C,cF/OuC,KE+O7B,KAAL,GAAiB,KAAtB,CD5O4C,MAAX,IAAf,C;O;KDLrD,C;4FAGA,yB;MAoOA ,6B;MApOA,wB;QAEqD,OCAA,cAAU,CD2O5B,cAAU,SAAL,GAAiB,GAAtB,CC3O4B,MAAK,GDAI,KCAO, KAAX,IAAf,C;O;KDFrD,C;4FAGA,yB;MA2OA,WAS6D,wB;MAT7D,+B;MA3OA,wB;QAEuD,OiBAA,eAAW, CjBkP7B,eAAW,oBAAL,SAAK,CAAL,UAAN,CiBIP6B,MAAK,KjBAI,KiBAO,KAAX,CAAhB,C;O;KjBFvD,C; 8FAIA,yB;MA6NA,6B;MA7NA,wB;QAEuD,OCMD,cAAU,CD8N7B,cAAU,SAAL,GAAiB,GAAtB,CC9N6B,M AAK,GAAY,CD8N9C,cApOwC,KAO09B,KAAL,GAAiB,GAAtB,CC9N8C,MAAZ,IAAf,C;O;KDRtD,C;8FAGA, yB;MA0NA,6B;MA1NA,wB;QAEwD,OCGF,cAAU,CD8N7B,cAAU,SAAL,GAAiB,GAAtB,CC9N6B,MAAK,G AAY,CC+N9C,cFIOyC,KEkO/B,KAAL,GAAiB,KAAtB,CD/N8C,MAAZ,IAAf,C;O;KDLtD,C;8FAGA,yB;MAuN A,6B;MAvNA,wB;QAEsD,OCAA,cAAU,CD8N7B,cAAU,SAAL,GAAiB,GAAtB,CC9N6B,MAAK,GDAK,KCA O,KAAZ,IAAf,C;O;KDFtD,C;8FAGA,yB;MA8NA,WAS6D,wB;MAT7D,+B;MA9NA,wB;QAEwD,OiBAA,eAA W,CjBqO9B,eAAW,oBAAL,SAAK,CAAL,UAAN,CiBrO8B,MAAK,UjBAK,KiBAO,KAAZ,CAAhB,C;O;KjBFx D,C;8FAIA,yB;MAGNA,6B;MAhNA,wB;QAEuD,OCMD,cAAe,YAAL,CDiN7B,cAAU,SAAL,GAAiB,GAAtB,C CjN6B,MAAK,EAAY,CDiN9C,cAvNwC,KAuN9B,KAAL,GAAiB,GAAtB,CCjN8C,MAAZ,CAAf,C;O;KDRtD,C ;8FAGA,yB;MA6MA,6B;MA7MA,wB;QAEwD,OCGF,cAAe,YAAL,CDiN7B,cAAU,SAAL,GAAiB,GAAtB,CCj N6B,MAAK,EAAY,CCKn9C,cFrNyC,KEqN/B,KAAL,GAAiB,KAAtB,CDiN8C,MAAZ,CAAf,C;O;KDLtD,C;8F AGA,yB;MA0MA,6B;MA1MA,wB;QAEsD,OCAA,cAAe,YAAL,CDiN7B,cAAU,SAAL,GAAiB,GAAtB,CCjN6B ,MAAK,EDAK,KCAO,KAAZ,CAAf,C;O;KDFtD,C;8FAGA,yB;MAiNA,WAS6D,wB;MAT7D,+B;MAjNA,wB;Q AEwD,OiBAA,eAAW,CjBwN9B,eAAW,oBAAL,SAAK,CAAL,UAAN,CiBxN8B,MAAK,UjBAK,KiBAO,KAAZ ,CAAhB,C;O;KjBFxD,C;0FAIA,yB;MAmMA,6B;MC7LA,4C;MDNA,wB;QAEqD,OCMD,WDoMjB,cAAU,SAAL,GAAiB,GAAtB,CCpMiB,EDoMjB,cA1MoC,KA0M1B,KAAL,GAAiB,GAAtB,CCpMiB,C;O;KDRpD,C;0FAG A,yB;MAGMA,6B;MC7LA,4C;MDHA,wB;QAEsD,OCGF,WDoMjB,cAAU,SAAL,GAAiB,GAAtB,CCpMiB,ECq MjB,cFxMqC,KEwM3B,KAAL,GAAiB,KAAtB,CDrMiB,C;O;KDLpD,C;0FAGA,yB;MA6LA,6B;MC7LA,4C;M

DAA,wB;QAEoD,OCAA,WDoMjB,cAAU,SAAL,GAAiB,GAAtB,CCpMiB,EDAkB,KCAiB,C;O;KDFpD,C;0FA GA,yB;MAoMA,WAS6D,wB;MAT7D,+B;MiBpMA,8C;MjBAA,wB;QAEsD,OiBAA,YjB2MjB,eAAW,oBAAL,S AAK,CAAL,UAAN,CiB3MiB,EjBAmB,KiBAnB,C;O;KjBFtD,C;0FAIA,yB;MA5LA,6B;MCxKA,kD;MDdA,wB; QAMqD,OCcD,cD2KjB,cAAU,SAAL,GAAiB,GAAtB,CC3KiB,ED2KjB,cAzLoC,KAyL1B,KAAL,GAAiB,GAAt B,CC3KiB,C;O;KDpBpD,C;0FAOA,yB;MA+KA,6B;MCxKA,kD;MDPA,wB;QAMsD,OCOF,cD2KjB,cAAU,SA AL,GAAiB,GAAtB,CC3KiB,EC4KjB,cFnLqC,KEmL3B,KAAL,GAAiB,KAAtB,CD5KiB,C;O;KDbpD,C;0FAOA, yB;MAwKA,6B;MCxKA,kD;MDAA,wB;QAMoD,OCAA,cD2KjB,cAAU,SAAL,GAAiB,GAAtB,CC3KiB,EDAK B,KCAiB,C;O;KDNpD,C;0FAOA,yB;MA2KA,WAS6D,wB;MAT7D,+B;MiB3KA,oD;MjBAA,wB;QAMsD,OiB AA,ejB8KjB,eAAW,oBAAL,SAAK,CAAL,UAAN,CiB9KiB,EjBAmB,KiBAnB,C;O;KjBNtD,C;oGAQA,yB;MAY JA,6B;MC7LA,4C;MDoCA,wB;QAMiD,OCxCG,WDoMjB,cAAU,SAAL,GAAiB,GAAtB,CCpMiB,EDoMjB,cA5 JqC,KA4J3B,KAAL,GAAiB,GAAtB,CCpMiB,C;O;KDKCpD,C;oGAOA,yB;MAkJA,6B;MC7LA,4C;MD2CA,wB; QAMkD,OC/CE,WDoMjB,cAAU,SAAL,GAAiB,GAAtB,CCpMiB,ECqMjB,cFtJsC,KEsJ5B,KAAL,GAAiB,KAA tB,CDrMiB,C;O;KDyCpD,C;oGAOA,yB;MA2IA,6B;MC7LA,4C;MDkDA,wB;QAMgD,OCtDI,WDoMjB,cAAU, SAAL,GAAiB,GAAtB,CCpMiB,EDsDmB,KCtDnB,C;O;KDgDpD,C;oGAOA,yB;MA8IA,WAS6D,wB;MAT7D,+ B;MiBpMA,8C;MjBsDA,wB;QAMkD,OiB1DI,YjB2MjB,eAAW,oBAAL,SAAK,CAAL,UAAN,CiB3MiB,EjB0D oB,KiB1DpB,C;O;KjBoDtD,C;0FAQA,yB;MA4HA,6B;MCxKA,kD;MDuOJ,0B;MAAA,+B;MA3LI,wB;QAQ6C, OA8LR,eAAW,OC5OI,cD2KjB,cAAU,SAAL,GAAiB,GAAtB,CC3KiB,ED2KjB,cA7H4B,KA6HIB,KAAL,GAAi B,GAAtB,CC3KiB,CAkLf,KD0DW,CAAX,C;O;KATMrC,C;0FASA,yB;MAMHA,6B;MCxKA,kD;MCwoJ,4B;M AAA,iC;MFnLI,wB;QAQ+C,OE5LR,gBAAy,QD7OC,cD2KjB,cAAU,SAAL,GAAiB,GAAtB,CC3KiB,EC4KjB,c FrH8B,KEqHpB,KAAL,GAAiB,KAAtB,CD5KiB,CA4Lb,KCiDY,CAAZ,C;O;KF9LvC,C;0FASA,yB;MA0GA,6B ;MCxKA,kD;MD8DA,wB;QAQ2C,OChES,cD2KjB,cAAU,SAAL,GAAiB,GAAtB,CC3KiB,EDgES,KChET,C;O; KDwDpD,C;0FASA,yB;MA2GA,WAS6D,wB;MAT7D,+B;MiB3KA,oD;MjBgEA,wB;QAQ6C,OiBIES,ejB8KjB,e AAW,oBAAL,SAAK,CAAL,UAAN,CiB9KiB,EjBkEU,KiBIEV,C;O;KjB0DtD,C;0EAUA,yB;MAAA,0B;MAAA, +B;MAAA,mB;QAM0C,sBAAW,OAAL,SAAK,KAAX,C;O;KAN1C,C;0EAQA,yB;MAAA,0B;MAAA,+B;MAA A,mB;QAM0C,sBAAW,OAAL,SAAK,KAAX,C;O;KAN1C,C;kGAQA,yB;MAAA,8C;MAuEA,6B;MAvEA,wB; QAE8D,0BA8E3B,cAAU,SAAL,GAAiB,GAAtB,CA9E2B,EA8E3B,cA9EoD,KA8E1C,KAAL,GAAiB,GAAtB,C A9E2B,C;O;KAF9D,C;0FAIA,yB;MAAA,+B;M4LxOJ,0B;M5LwOI,wB;QAEEmD,sB4LvOgC,O5LuO1B,IAAK,K 4LvOX,G5LuOoB,KAAM,K4LvOM,C5LuOhC,C;O;KAFnD,C;wFAGA,yB;MAAA,+B;M4LtoJ,0B;M5LsOI,wB; QAEkD,sB4LrO+B,O5LqOzB,IAAK,K4LrOX,G5LqOmB,KAAM,K4LrOM,C5LqO/B,C;O;KAFID,C;0FAGA,yB; MAAA,+B;M4LpOJ,0B;M5LoOI,wB;QAEEmD,sB4LnOgC,O5LmO1B,IAAK,K4LnOX,G5LmOoB,KAAM,K4Ln OM,C5LmOhC,C;O;KAFnD,C;0EAGA,yB;MAAA,+B;M4LlOJ,0B;M5LkOI,mB;QAEiC,sB4LjOqB,OAAP,C5Li OR,S4LjOe,C5LiOrB,C;O;KAFjC,C;gFAIA,Y;MASmC,gB;K;kFACnC,yB;M4LlOJ,4B;M5L0OI,mB;QASqC,O4 LhPiD,Q5LgP5C,S4LhPY,G5LgPE,G4LhP8B,C;O;K5LuOtF,C;8EAUA,Y;MASiC,OAAL,SAAL,GAAiB,G;K;gF ACID,yB;MAAA,WASqD,wB;MATrD,mB;QASmC,OAAL,oBAAL,SAAK,CAAL,U;O;KATnC,C;kFAWA,Y;M AEqC,W;K;oFAFrC,yB;MAAA,iC;M4L5QJ,4B;M5L4QI,mB;QASuC,uB4LIR+C,Q5LkRnC,S4LIRG,G5LkRW,G 4LIRqB,C5LkR/C,C;O;KATvC,C;gFAUA,yB;MAAA,6B;MAAA,mB;QASmC,qBAAU,SAAL,GAAiB,GAAtB,C; O;KATnC,C;kFAUA,yB;MAAA,WAS6D,wB;MAT7D,+B;MAAA,mB;QASqC,sBAAW,oBAAL,SAAK,CAAL,U AAN,C;O;KATrC,C;kFAWA,Y;MAMqC,OApDC,SAAL,GAAiB,G;K;oFAqDID,Y;MAMuC,OA3DD,SAAL,GA AiB,G;K;+BA6DID,Y;MAAyC,OAAQ,CA7DX,SAAL,GAAiB,GA6DD,Y;K;+BA1UrD,Y;MAAA,c;MAG4D,q D;MAH5D,a;K;6BAAA,iB;MAAA,2IAG4D,oCAH5D,G;K;wEA8UA,yB;MAAA,+B;MAAA,4B;QAU0C,sBAAM ,SAAN,C;O;KAV1C,C;0EAWA,yB;MAAA,0B;MAAA,+B;MAAA,4B;QAW2C,sBAAW,OAAL,SAAK,CAAX,C; O;KAX3C,C;0EAYA,yB;MAAA,0B;MAAA,+B;MAAA,4B;QAWyC,sBAAW,OAAL,SAAK,CAAX,C;O;KAXzC ,C;0EAYA,yB;MAAA,0B;MAAA,+B;MAAA,4B;QAW0C,sBAAW,OAAL,SAAK,SAAX,C;O;KAX1C,C;Igc9W A,6B;MACqB,sB;K;uCAKjB,iB;MAM6C,OhCyUP,UgCzUO,aAAQ,KAAR,ChCyUP,C;K;uCGCvUtC,wB;MAOI, aAAQ,KAAR,IAAiB,KhCiOc,K;K;K;FgC7NL,Y;MAAQ,OAAA,YAAQ,O;K;oCAE9C,Y;MAC8E,+BAAS,YAAT, C;K;IAGxD,oC;MAAiC,wB;MAAhC,oB;MACnB,eAAoB,C;K;4CACpB,Y;MAAyB,sBAAQ,YAAM,O;K;8CACv C,Y;MAAyD,Q;MAA9B,IAAI,eAAQ,YAAM,OAAIB,C;QAAA,OhCmTO,UgCnTiB,aAAM,mBAAN,EAAM,2B AAN,OhCmTjB,C;;QgCnT+C,MAAM,2BAAuB,YAAM,WAA7B,C;K;;0CAG3F,mB;MAIS,Q;MAAL,IAAI,eAA

C,0EAAD,QAAJ,C;QAAiC,OAAO,K;MAExC,OAAe,WAAR,YAAQ,EAAS,OhC2MO,KgC3MhB,C;K;+CAGnB,  
oB;MACY,Q;MAA2B,gBAA3B,gE;MAA2B,c;;Qd0nDvB,U;QADhB,IAAI,wCAAsB,mBAA1B,C;UAAqC,aAAO,  
I;UAAp,e;SACrB,6B;QAAhB,OAAgB,gBAAhB,C;UAAgB,2B;Uc1nD6B,2Bd0nDR,Oc1nDQ,Q;UAAA,W;YAAu  
B,oBAAR,YAAQ,Ed0nD/B,OIBn7CF,KgCvMiC,C;Wd0nD9C,IAAI,OAAJ,C;YAAyB,aAAO,K;YAAP,e;;QAC/C,  
aAAO,I;;Mc3nDH,iB;K;mCAGJ,Y;MAAkC,OAAA,IAAK,QAAQ,OAAb,KAAqB,C;K;;IA/CvD,sC;MAAA,oD;  
MACgC,uBAAK,cAAU,IAAV,CAAL,C;MADhC,Y;K;;;oCAPJ,Y;MAAA,OAKqB,qDALrB,M;K;oCAAA,Y;MA  
AA,c;MAKqB,wD;MALrB,a;K;kCAAA,iB;MAAA,2IAKqB,0CALrB,G;K;gFAyDA,yB;MAAA,yC;MAWSc,yC;  
QAAA,wB;UAAW,OAAA,aAAK,KAAL,ChCsLV,K;S;O;MgCjMvC,6B;QAWI,OAAO,oBAAW,+BAAU,IAAV,  
GAAgB,uBAAhB,CAAX,C;O;KAXX,C;kFAcA,oB;MAGqE,e;K;I/BtE7C,oB;MAEpB,4B;MAFuD,gB;K;IAEvD,0  
B;MAAA,8B;MACI,iBAGmC,SAAK,CAAL,C;MAEnC,iBAGmC,SAAK,EAAL,C;MAEnC,kBAGmC,C;MAEnC,  
iBAGkC,E;K;;IANBtC,sC;MAAA,qC;QAAA,oB;OAAA,8B;K;oGAsBA,yB;MD2QA,6B;MC3PA,8C;MAhBA,wB  
;QAM0D,OAIbQ,YAAy,IAAK,KAAjB,EAA6B,CD6P5D,cC9QsC,KD8Q5B,KAAL,GAAiB,GAAtB,CC7P4D,M  
AA7B,C;O;KAvBIE,C;oGAQA,yB;MCoQA,6B;MD5PA,8C;MARA,wB;QAM2D,OASO,YAAy,IAAK,KAAjB,E  
AA6B,CC8P5D,cDvQuC,KCuQ7B,KAAL,GAAiB,KAAtB,CD9P4D,MAA7B,C;O;KAFIE,C;gGAQA,yB;MAAA,8  
C;MAAA,wB;QAOKE,mBAAy,IAAK,KAAjB,EAAuB,KAAM,KAA7B,C;O;KAPIE,C;oGASA,yB;MAGRA,kBA  
S6D,sB;MAT7D,+B;MgBjRA,gD;MhBCA,wB;QAM0D,OgBAS,aAAkB,ChBmRhD,eAAW,oBAAL,SAAK,CAA  
L,iBAAN,CgBnRgD,MAAiB,EhBAgB,KgBAc,KAA9B,C;O;KhBNnE,C;0FAQA,yB;MD0OA,6B;MC1OA,wB;Q  
AEsD,OAMD,cAAK,IAAK,KAAL,GAAW,CD2O5C,cCjP6B,KDiPnB,KAAL,GAAiB,GAAtB,CC3O4C,MAAX,I  
AAf,C;O;KARrD,C;0FAGA,yB;MCwOA,6B;MDxOA,wB;QAEuD,OAGF,cAAK,IAAK,KAAL,GAAW,CC4O5C,  
cD/O8B,KC+OpB,KAAL,GAAiB,KAAtB,CD5O4C,MAAX,IAAf,C;O;KALrD,C;0FAGA,yB;MAAA,6B;MAAA,  
wB;QAEqD,qBAAK,IAAK,KAAL,GAAL,KAAM,KAAX,IAAf,C;O;KAFrD,C;0FAGA,yB;MA+PA,kBAS6D,sB;  
MAT7D,+B;MA/PA,wB;QAEuD,OgBAA,eAAW,ChBsQ7B,eAAW,oBAAL,SAAK,CAAL,iBAAN,CgBtQ6B,MA  
AK,KhBAL,KgBAO,KAAX,CAAhB,C;O;KhBFvD,C;4FAIA,yB;MD6NA,6B;MC7NA,wB;QAEuD,OAMD,cAAK  
,IAAK,KAAL,GAAY,CD8N9C,cCpO+B,KDoOrB,KAAL,GAAiB,GAAtB,CC9N8C,MAAZ,IAAf,C;O;KARtD,C;  
4FAGA,yB;MC2NA,6B;MD3NA,wB;QAEwD,OAGF,cAAK,IAAK,KAAL,GAAY,CC+N9C,cDIogC,KCkOtB,K  
AAL,GAAiB,KAAtB,CD/N8C,MAAZ,IAAf,C;O;KALtD,C;4FAGA,yB;MAAA,6B;MAAA,wB;QAEsD,qBAAK,I  
AAK,KAAL,GAAM,KAAM,KAAX,IAAf,C;O;KAFtD,C;4FAGA,yB;MAkPA,kBAS6D,sB;MAT7D,+B;MAIPA,  
wB;QAEwD,OgBAA,eAAW,ChByP9B,eAAW,oBAAL,SAAK,CAAL,iBAAN,CgBzP8B,MAAK,UhBAK,KgBAO  
,KAAZ,CAAhB,C;O;KhBFxD,C;4FAIA,yB;MDgNA,6B;MChNA,wB;QAEuD,OAMD,cAAe,YAAV,IAAK,KAA  
K,EAAY,CDiN9C,cCvN+B,KDuNrB,KAAL,GAAiB,GAAtB,CCjN8C,MAAZ,CAAf,C;O;KARtD,C;4FAGA,yB;  
MC8MA,6B;MD9MA,wB;QAEwD,OAGF,cAAe,YAAV,IAAK,KAAL,EAAY,CCkN9C,cDrNgC,KCqNtB,KAAL  
,GAAiB,KAAtB,CDiN8C,MAAZ,CAAf,C;O;KALtD,C;4FAGA,yB;MAAA,6B;MAAA,wB;QAEsD,qBAAe,YAA  
V,IAAK,KAAL,EAAM,KAAM,KAAX,CAAf,C;O;KAFtD,C;4FAGA,yB;MAqOA,kBAS6D,sB;MAT7D,+B;MAr  
OA,wB;QAEwD,OgBAA,eAAW,ChB4O9B,eAAW,oBAAL,SAAK,CAAL,iBAAN,CgB5O8B,MAAK,UhBAK,Kg  
BAO,KAAZ,CAAhB,C;O;KhBFxD,C;wFAIA,yB;MDmMA,6B;MC7LA,4C;MANA,wB;QAEqD,OAMD,WAAW,  
IAAX,EDoMjB,cC1M2B,KD0MjB,KAAL,GAAiB,GAAtB,CCpMiB,C;O;KARpD,C;wFAGA,yB;MCiMA,6B;MD  
9LA,4C;MAHA,wB;QAEsD,OAGF,WAAW,IAAX,ECqMjB,cDxM4B,KCwMIB,KAAL,GAAiB,KAAtB,CDrMiB  
,C;O;KALpD,C;wFAGA,yB;MAAA,4C;MAAA,wB;QAEoD,kBAAW,IAAX,EAiB,KAAjB,C;O;KAFpD,C;wFA  
GA,yB;MAwNA,kBAS6D,sB;MAT7D,+B;MgBxNA,8C;MhBAA,wB;QAEsD,OgBAA,YhB+NjB,eAAW,oBAAL,  
SAAK,CAAL,iBAAN,CgB/NiB,EhBAmb,KgBAnB,C;O;KhBFtD,C;wFAIA,yB;MDsLA,6B;MCxKA,kD;MAAdA,  
wB;QAMqD,OAcD,cAAc,IAAd,ED2KjB,cCzL2B,KDyLjB,KAAL,GAAiB,GAAtB,CC3KiB,C;O;KApBpD,C;wF  
AOA,yB;MCgLA,6B;MDzKA,kD;MAPA,wB;QAMsD,OAOf,cAAc,IAAd,EC4KjB,cDnL4B,KCmLiB,KAAL,GA  
AiB,KAAtB,CD5KiB,C;O;KAbpD,C;wFAOA,yB;MAAA,kD;MAAA,wB;QAMoD,qBAAc,IAAd,EAAoB,KAApB  
,C;O;KANpD,C;wFAOA,yB;MA+LA,kBAS6D,sB;MAT7D,+B;MgB/LA,oD;MhBAA,wB;QAMsD,OgBAA,ehBk  
MjB,eAAW,oBAAL,SAAK,CAAL,iBAAN,CgBlMiB,EhBAmb,KgBAnB,C;O;KhBNtD,C;kGAQA,yB;MDyJA,6  
B;MC7LA,4C;MAoCA,wB;QAMiD,OAxCg,WAAW,IAAX,EDoMjB,cC5J4B,KD4JIB,KAAL,GAAiB,GAAtB,C  
CpMiB,C;O;KAKpD,C;kGAOA,yB;MCMJA,6B;MD9LA,4C;MA2CA,wB;QAMkD,OA/CE,WAAW,IAAX,ECq  
MjB,cDtJ6B,KCsJnB,KAAL,GAAiB,KAAtB,CDrMiB,C;O;KAYCpD,C;kGAOA,yB;MAIDA,4C;MAkDA,wB;QA

MgD,OAtDI,WAAW,IAAX,EAsDA,KAtDA,C;O;KAgDpD,C;kGAOA,yB;MAkKA,kBAS6D,sB;MAT7D,+B;Mg  
BxNA,8C;MhBsDA,wB;QAMkD,OgB1DI,YhB+NjB,eAAW,oBAAL,SAAK,CAAL,iBAAN,CgB/NiB,EhB0DoB,  
KgB1DpB,C;O;KhBoDtD,C;wFAQA,yB;MD4HA,6B;MCxKA,kD;MDuOJ,0B;MAAA,+B;MC3LI,wB;QAQ6C,O  
D8LR,eAAW,OC5OI,cAAc,IAAd,ED2KjB,c7HmB,KD6HT,KAAL,GAAiB,GAAtB,CC3KiB,CAkLf,KD0DW,C  
AAX,C;O;KcTMrC,C;wFASA,yB;MCoHA,6B;MDzKA,kD;MCwOJ,4B;MAAA,iC;MDnLI,wB;QAQ+C,OCsLR,g  
BAAY,QD7OC,cAAc,IAAd,EC4KjB,cDrHqB,KCqHX,KAAL,GAAiB,KAAtB,CD5KiB,CA4Lb,KCiDY,CAAZ,C;  
O;KD9LvC,C;wFASA,yB;MA9DA,kD;MA8DA,wB;QAQ2C,OAhES,cAAc,IAAd,EAgEL,KAhEK,C;O;KAwDpD  
,C;wFASA,yB;MA+HA,kBAS6D,sB;MAT7D,+B;MgB/LA,oD;MhBgEA,wB;QAQ6C,OgBIES,ehBkMjB,eAAW,o  
BAAL,SAAK,CAAL,iBAAN,CgBIMiB,EhBkEU,KgBIEV,C;O;KhB0DtD,C;wEAUA,yB;MAAA,6B;MAAA,mB;  
QAMyC,qBAAK,SAAK,QAAY,C;O;KANzC,C;wEAQA,yB;MAAA,6B;MAAA,mB;QAMyC,qBAAK,SAAK,QA  
AV,C;O;KANzC,C;gGAQA,yB;MAAA,8C;MAAA,wB;QAE6D,0BAAU,IAAV,EAAGB,KAhB,C;O;KAF7D,C;  
wFAIA,yB;MAAA,6B;MAAA,2B;QAOMD,qBAAK,aAAS,QAAD,C;O;KAPnD,C;wFASA,yB;MAAA,6B;MAAA,  
2B;QAOMD,qBAAK,cAAU,QAAf,C;O;KAPnD,C;wFASA,yB;MAAA,6B;MAAA,wB;QAEiD,qBAAK,IAAK,KA  
AL,GAAc,KAAM,KAAzB,C;O;KAFjD,C;wFAGA,yB;MAAA,6B;MAAA,wB;QAEgD,qBAAK,IAAK,KAAL,GA  
Aa,KAAM,KAAxB,C;O;KAFhD,C;wFAGA,yB;MAAA,6B;MAAA,wB;QAEiD,qBAAK,IAAK,KAAL,GAAc,KA  
AM,KAAzB,C;O;KAFjD,C;wEAGA,yB;MAAA,6B;MAAA,mB;QAEgC,qBAAU,CAAL,SAAL,C;O;KAFhC,C;8  
EAIA,yB;MAAA,0B;MAAA,mB;QAUmC,OAAK,OAAAL,SAAK,C;O;KAVxC,C;gFAWA,yB;MAAA,4B;MAAA,  
mB;QAUqC,OAAK,QAAL,SAAK,C;O;KAV1C,C;4EAWA,Y;MASiC,gB;K;8EACjC,yB;MAAA,kBASqD,sB;MA  
TrD,mB;QASmC,OAAK,oBAAL,SAAK,CAAL,iB;O;KATnC,C;gFAWA,yB;MDwDJ,0B;MAAA,+B;MCxDI,mB;  
QASqC,OD0DA,eAAW,OC1DX,SD0DW,CAAX,C;O;KcNerC,C;kFAUA,yB;MC+CJ,4B;MAAA,iC;MD/CI,mB;  
QASuC,OCiDA,gBAAY,QDjDZ,SCiDY,CAAZ,C;O;KD1DvC,C;8EAUA,Y;MAEmC,W;K;gFACnC,yB;MAAA,k  
BAS6D,sB;MAT7D,+B;MAAA,mB;QASqC,sBAAW,oBAAL,SAAK,CAAL,iBAAN,C;O;KATrC,C;gFAWA,yB;  
MASA,gD;MATA,mB;QAQqC,OAoe,aAAa,SAAb,C;O;KAFvC,C;kFASA,yB;MAAA,gD;MAAA,mB;QAMuC,o  
BAAa,SAAb,C;O;KANvC,C;8BAQA,Y;MAAYC,OArDD,oBAAL,SAAK,CAAL,iBAqDe,W;K;,,,;8BAhWtD,Y;M  
AAA,c;MAG2D,qD;MAH3D,a;K;4BAAA,iB;MAAA,2IAG2D,oCAH3D,G;K;sEAoWA,yB;MAAA,6B;MAAA,4B  
;QAWwC,qBAAU,SAAV,C;O;KAXxC,C;wEAYA,yB;MAAA,6B;MAAA,4B;QAWyC,qBAAU,SAAV,C;O;KAX  
zC,C;wEAYA,yB;MAAA,6B;MAAA,4B;QAUuC,qBAAK,SAAL,C;O;KAVvC,C;wEAWA,yB;MAAA,6B;MAAA  
,4B;QAWwC,qBAAK,SAAK,QAAY,C;O;KAXxC,C;uEAaA,yB;MAAA,gD;MAAA,4B;QASyC,oBAAKB,SAAlB,  
C;O;KATzC,C;wEAUA,yB;MAAA,gD;MAAA,4B;QAS0C,oBAAa,SAAb,C;O;KAT1C,C;Igc3ZA,4B;MACqB,sB  
;K;sCAKjB,iB;MAM4C,OhCuXT,SgCvXS,aAAQ,KAAR,ChCuXT,C;K;sCgCrXnC,wB;MAOI,aAAQ,KAAR,IAA  
iB,KhCyQY,K;K;iFgCrQH,Y;MAAQ,OAAA,YAAQ,O;K;mCAE9C,Y;MAC6E,8BAAS,YAAT,C;K;IAGvD,mC;  
MAAgC,uB;MAA/B,oB;MACnB,eAAoB,C;K;2CACpB,Y;MAAYB,sBAAQ,YAAM,O;K;4CACvC,Y;MAAwD,Q;  
MAA9B,IAAI,eAAQ,YAAM,OAAIB,C;QAAA,OhCiWK,SgCjWmB,aAAM,mBAAN,EAAM,2BAAN,OhCiWnB,  
C;;QgCjWgD,MAAM,2BAAuB,YAAM,WAA7B,C;K;;yCAGzF,mB;MAIS,Q;MAAL,IAAI,eAAC,0EAAD,0AAJ,  
C;QAAgC,OAAO,K;MAEvC,OAAe,WAAR,YAAQ,EAAS,OhCmPK,KgCnPd,C;K;8CAGnB,oB;MACY,Q;MAA  
2B,gBAA3B,gE;MAA2B,c;;Qf0nDvB,U;QADhB,IAAI,wCAAsB,mBAA1B,C;UAAqC,aAAO,I;UAAP,e;SACrB,6  
B;QAAhB,OAAgB,gBAAhB,C;UAAgB,2B;Ue1nD6B,2Bf0nDR,Oe1nDQ,O;UAAA,W;YAAsB,oBAAR,YAAQ,E  
f0nD9B,OjB34CJ,KgC/OkC,C;Wf0nD7C,IAAI,0AAJ,C;YAAyB,aAAO,K;YAAP,e;;QAC/C,aAAO,I;;Me3nDH,i  
B;K;kCAGJ,Y;MAAkC,OAAA,IAAK,QAAQ,OAAb,KAAqB,C;K;;IA/CvD,qC;MAAA,mD;MACgC,sBAAK,eAA  
S,IAAT,CAAL,C;MADhC,Y;K;;;mCAPJ,Y;MAAA,OAKqB,oDALrB,M;K;mCAAA,Y;MAAA,c;MAKqB,wD;M  
ALrB,a;K;iCAAA,iB;MAAA,2IAKqB,0CALrB,G;K;8EAyDA,yB;MAAA,uC;MAWoC,wC;QAAA,wB;UAAW,O  
AAA,aAAK,KAAL,ChC8NV,K;S;O;MgCzOrC,6B;QAWI,OAAO,mBAAU,gCAAS,IAAT,GAAe,sBAAf,CAAV,  
C;O;KAXX,C;gFACa,oB;MAGkE,e;K;I4LnE5C,wC;MASBIB,iC;MATBsD,2BAAgB,KAhB,EAAuB,YAAvB,EA  
AqC,CAArC,C;K;kFAC7B,Y;MAAQ,iB;K;yFACD,Y;MAAQ,gB;K;2CAExC,iB;MAA8C,W5NwCoB,Y4NxCpB,  
U5NwCqC,KAAjB,E4NxCX,K5NwCwC,KAA7B,C4NxCpB,K;MAAA,S;QAAkB,O5NwCE,Y4NxCf,K5NwCmB  
,KAAjB,E4NxC0,S5NwCsB,KAA7B,C4NxCf,K;OAAIB,W;K;kCAE9C,Y;MAKkC,O5NiCgC,Y4NjChC,U5NiCi  
D,KAAjB,E4NjCxB,S5NiCqD,KAA7B,C4NjChC,I;K;iCAEIC,iB;MAEY,UAAwB,M;MADhC,2CAAuB,kBAAa,  
KAAM,UAAAnB,KACf,2CAAS,KAAM,MAAf,cAAwB,6CAAQ,KAAM,KAAAd,QAAxB,CADe,CAAvB,C;K;mCA

GJ, Y; MACI, OAAI, cAAJ, GAAe, EAAf, GAAwB, MAAK, U5NyQA, K4NzQL, QAAqB, S5NyQhB, K4NzQL, I; K; mC  
AE5B, Y; MAAkC, OAAE, UAAF, qBAAU, S; K; IAE5C, +B; MAAA, mC; MACI, aAC8B, cAAU, 4BAAK, UAAF, EAAO  
B, 4BAAK, UAA/B, C; K; ; IAFIC, 2C; MAAA, 0C; QAAA, yB; OAAA, mC; K; ; IAYJ, oD; MA4CI, uC; MATCI, IAAI, SAA  
Q, CAAZ, C; QAAuB, MAAa, gCAAyB, wBAAzB, C; MACpC, IAAI, SAAQ, WAAZ, C; QAA2B, MAAa, gCAAyB, wEA  
AzB, C; MAG5C, aAGyB, K; MAEZB, YAGwB, 4BAA0B, KAA1B, EAAiC, YAAjC, EAA+C, IAA/C, C; MAExB, YAGu  
B, I; K; yCAEvB, Y; MAAgD, mCAAwB, UAAxB, EAA+B, SAA/B, EAAqC, SAARc, C; K; wCAEhD, Y; MAMqC, OAAI,  
YAAO, CAAX, G5NvB6B, Y4NuBf, U5NvBgC, KAAjB, E4NuBP, S5NvBoC, KAA7B, C4NuBf, IAA, G5NvB6B, Y4N  
uBG, U5NvBc, KAAjB, E4NuBW, S5NvBk, KAA7B, C4NuBG, I; K; uCAErE, iB; MAEY, UAAwB, M; MADhC, iDAA  
6B, kBAAa, KAAM, UAAAnB, KACrB, 2CAAS, KAAM, MAAf, cAAwB, 6CAAQ, KAAM, KAA, QAAxB, KAA8C, cA  
AQ, KAAM, KADvC, CAA7B, C; K; yCAGJ, Y; MACI, OAAI, cAAJ, GAAe, EAAf, GAAwB, OAAAM, MAAK, U5NiNN,  
K4NjNC, QAAqB, S5NiNtB, K4NjNC, IAAN, SAAgD, SAAhD, I; K; yCAE5B, Y; MAAkC, OAAI, YAAO, CAAX, GAA  
gB, UAAF, qBAAU, SAAV, cAAqB, SAAnc, GAAgD, UAAF, 2BAAgB, SAAhB, eAA4B, CAAC, SAAD, IAA5B, C; K; I  
AEhF, qC; MAAA, yC; K; kEACI, sC; MAQ2F, 2BAAgB, UAAhB, EAA4B, QAA5B, EAA5C, IAAc, C; K; ; IAT/F, iD; M  
AAA, gD; QAAA, +B; OAAA, yC; K; ; IAoBiC, oD; MAAuC, uB; MACxE, sBAA2B, I; MAC3B, iBAAmC, OAAO, CAA1  
C, G5NxDe, Y4NwDrB, K5NxDsC, KAAjB, E4NwDZ, I5NxDyC, KAA7B, C4NwDrB, KAA7C, G5NxDe, Y4NwDF,  
K5NxDb, KAAjB, E4NwDO, I5NxDsB, KAA7B, C4NwDF, K; MACH, c5N2RmC, S4N3RhB, I5N2RgB, C; M4N1Rn  
C, cAAuB, cAAJ, GAAa, KAAb, GAAwB, mB; K; gDAE3C, Y; MAAkC, qB; K; iDAEIC, Y; MACI, YAAy, W; MACZ, IA  
AI, 6BAAS, mBAAT, QAAJ, C; QACI, IAAI, CAAC, cAAL, C; UAAc, MAAa, 6B; QAC3B, iBAAU, K; ; QAEV, c5NID6C,  
S4NkD7C, W5NIDuD, KAAK, G4NkDpD, W5NID+D, KAAX, IAAf, C; ; M4NoDjD, OAAO, K; K; ; IC3Hf, yB; K; mCAII,  
Y; MAA4B, uB; K; ; IAMhC, 0B; K; oCAII, Y; MAA4B, wB; K; ; IAMhC, wB; K; kCAII, Y; MAA4B, sB; K; ; IAMhC, yB; K; m  
CAII, Y; MAA4B, uB; K; ; I7M5BP, qB; MAERB, 6B; MAFwD, gB; K; IAExD, 2B; MAAA, +B; MACI, iBAGoC, a; MAEPc,  
iBAGoC, c; MAEPc, kBAGmC, C; MAEnC, iBAGkC, E; K; ; IAnBtC, uC; MAAA, sC; QAAA, qB; OAAA, +B; K; sGAsBA  
, yB; MjBqRA, WAS6D, wB; MAT7D, +B; MiB7PA, gD; MAxB, wB; QAM0D, OAYBS, aAAa, IAAK, KAAIB, EAA8B,  
CjB+P5D, eAAW, oBiBxRyB, KjBwR9B, KAAK, CAAL, UAAN, CiB/P4D, MAA9B, C; O; KA/BnE, C; sGAQA, yB; Mf8  
QA, aAS6D, 0B; MAT7D, +B; Me9PA, gD; MAhBA, wB; QAM2D, OAiBQ, aAAa, IAAK, KAAIB, EAA8B, CfgQ5D, eAA  
W, oBejR0B, Kfir/B, KAAK, CAAL, YAAN, CehQ4D, MAA9B, C; O; KAvBnE, C; sGAQA, yB; MhByRA, kBAS6D, sB;  
MAT7D, +B; MgBjRA, gD; MARA, wB; QAMyD, OASU, aAAa, IAAK, KAAIB, EAA8B, ChBmR5D, eAAW, oBgB5Rw  
B, KhB4R7B, KAAK, CAAL, iBAAN, CgBnR4D, MAA9B, C; O; KAFnE, C; kGAQA, yB; MAAA, gD; MAAA, wB; QAO  
mE, oBAAa, IAAK, KAAIB, EAAwB, KAAM, KAA9B, C; O; KAPnE, C; 4FASA, yB; MjBoPA, WAS6D, wB; MAT7D, +B  
; MiBpPA, wB; QAEuD, OASA, eAAM, IAAK, KAAK, KAAW, CjBkP7C, eAAW, oBiB3PiB, KjB2PtB, KAAK, CAAL,  
UAAN, CiBIP6C, MAAX, CAAhB, C; O; KAXvD, C; 4FAGA, yB; MfkPA, aAS6D, 0B; MAT7D, +B; MeIP, wB; QAEwD  
, OAMD, eAAM, IAAK, KAAK, KAAW, CfmP7C, eAAW, oBezPkB, KfyPvB, KAAK, CAAL, YAAN, CenP6C, MAAX,  
CAAhB, C; O; KARvD, C; 4FAGA, yB; MhBkQA, kBAS6D, sB; MAT7D, +B; MgBIQA, wB; QAEsD, OAGC, eAAM, IAA  
K, KAAK, KAAW, ChBsQ7C, eAAW, oBgBzQgB, KhByQrB, KAAK, CAAL, iBAAN, CgBtQ6C, MAAX, CAAhB, C; O;  
KALvD, C; 4FAGA, yB; MAAA, +B; MAAA, wB; QAEuD, sBAAM, IAAK, KAAK, KAAK, KAAM, KAAX, CAAhB, C;  
O; KAFvD, C; 8FAIA, yB; MjBuOA, WAS6D, wB; MAT7D, +B; MiBvOA, wB; QAEwD, OASA, eAAM, IAAK, KAAK, U  
AAy, CjBqO/C, eAAW, oBiB9OmB, KjB8OxB, KAAK, CAAL, UAAN, CiBrO+C, MAAZ, CAAhB, C; O; KAXxD, C; 8F  
AGA, yB; MfqOA, aAS6D, 0B; MAT7D, +B; MerOA, wB; QAEyD, OAMD, eAAM, IAAK, KAAK, UAAy, CfsO/C, eAA  
W, oBe5OoB, Kf4OzB, KAAK, CAAL, YAAN, CetO+C, MAAZ, CAAhB, C; O; KARxD, C; 8FAGA, yB; MhBqPA, kBAS  
6D, sB; MAT7D, +B; MgBrPA, wB; QAEuD, OAGC, eAAM, IAAK, KAAK, UAAy, ChByP/C, eAAW, oBgB5PkB, KhB4  
PvB, KAAK, CAAL, iBAAN, CgBzP+C, MAAZ, CAAhB, C; O; KALxD, C; 8FAGA, yB; MAAA, +B; MAAA, wB; QAEw  
D, sBAAM, IAAK, KAAK, UAAAM, KAAM, KAAZ, CAAhB, C; O; KAFxD, C; 8FAIA, yB; MjB0NA, WAS6D, wB; MAT7  
D, +B; MiB1NA, wB; QAEwD, OASA, eAAM, IAAK, KAAK, UAAy, CjBwN/C, eAAW, oBiBjOmB, KjBiOxB, KAAK,  
CAAL, UAAN, CiBxN+C, MAAZ, CAAhB, C; O; KAXxD, C; 8FAGA, yB; MfwnA, aAS6D, 0B; MAT7D, +B; MexNA, w  
B; QAEyD, OAMD, eAAM, IAAK, KAAK, UAAy, CfyN/C, eAAW, oBe/NoB, Kf+NzB, KAAK, CAAL, YAAN, CezN+  
C, MAAZ, CAAhB, C; O; KARxD, C; 8FAGA, yB; MhBwOA, kBAS6D, sB; MAT7D, +B; MgBxOA, wB; QAEuD, OAGC,  
eAAM, IAAK, KAAK, UAAy, ChB4O/C, eAAW, oBgB/OkB, KhB+OvB, KAAK, CAAL, iBAAN, CgB5O+C, MAAZ, C  
AAhB, C; O; KALxD, C; 8FAGA, yB; MAAA, +B; MAAA, wB; QAEwD, sBAAM, IAAK, KAAK, UAAAM, KAAM, KAAZ



yD,Q;MAA9B,IAAI,eAAQ,YAAM,OAAIB,C;QAAA,OjBgXO,UiBhXiB,aAAM,mBAAN,EAAM,2BAAN,OjBgX  
jB,C;;QiBhX+C,MAAM,2BAAuB,YAAM,WAA7B,C;K;;0CAG3F,mB;MAIS,Q;MAAL,IAAI,eAAC,0EAAD,QA  
AJ,C;QAAiC,OAAO,K;MAExC,OAAe,WAAR,YAAQ,EAAS,OjB8PO,KiB9PhB,C;K;+CAGnB,oB;MACY,Q;MA  
A2B,gBAA3B,gE;MAA2B,c;;QhB0nDvB,U;QADhB,IAAI,wCAAsB,mBAA1B,C;UAAqC,aAAO,I;UAAP,e;SACr  
B,6B;QAAhB,OAAgB,gBAAhB,C;UAAgB,2B;UgB1nD6B,2BhB0nDR,OgB1nDQ,Q;UAAA,W;YAAuB,oBAAR,  
YAAQ,EhB0nD/B,ODh4CF,KiB1PiC,C;WhB0nD9C,IAAI,OAAJ,C;YAAyB,aAAO,K;YAAP,e;;QAC/C,aAAO,I;;  
MgB3nDH,iB;K;mCAGJ,Y;MAAkC,OAAA,IAAK,QAAQ,OAAb,KAAqB,C;K;;IA/CvD,sC;MAAA,oD;MACgC,  
uBAAK,iBAAU,IAAV,CAAL,C;MADhC,Y;K;;;oCAPJ,Y;MAAA,OAKqB,qDALrB,M;K;oCAA,Y;MAAA,c;M  
AKqB,wD;MALrB,a;K;kCAAA,iB;MAAA,2IAKqB,0CALrB,G;K;gFAyDA,yB;MAAA,yC;MAWsC,yC;QAAA,w  
B;UAAW,OAAA,aAAK,KAAL,CjByOV,K;S;O;MiBpPvC,6B;QAWI,OAAO,oBAAW,kBAAU,IAAV,EAAGB,uB  
AAhB,CAAX,C;O;KAXX,C;kFAcA,oB;MAGqE,e;K;I6LnE9C,2C;MAsBnB,kC;MatByD,4BAAiB,KAAjB,EAA  
wB,YAAxB,K;K;qFAC/B,Y;MAAQ,iB;K;4FACD,Y;MAAQ,gB;K;8CAEzC,iB;MAA+C,W9MgDoB,a8MhDpB,U  
9MgDsC,KAAIB,E8MhDX,K9MgDyC,KAA9B,C8MhDpB,K;MAAA,S;QAAkB,O9MgDE,a8MhDF,K9MgDoB,K  
AAIB,E8MhDO,S9MgDuB,KAA9B,C8MhDF,K;OAAIB,W;K;qCAE/C,Y;MAKkC,O9MyCiC,a8MzCjC,U9MyCm  
D,KAAIB,E8MzCzB,S9MyCuD,KAA9B,C8MzCjC,I;K;oCAEiC,iB;MAEY,UAAwB,M;MADhC,8CAAwB,kBAA  
a,KAAM,UAAAnB,KACHB,2CAAS,KAAM,MAAf,cAAwB,6CAAQ,KAAM,KAAd,QAAxB,CADgB,CAAXB,C;K;  
sCAGJ,Y;MACI,OAAI,cAAJ,GAAe,EAAf,GAAwB,M9M0QK,CARckB,U8MrOjB,U9MqO4B,KAAL,KAAoB,C  
AVzB,U8M3NP,U9M2Na,yB8M3NH,E9M2NG,CAAN,CAUyB,MAApB,CAAN,CAqCIB,MAAK,Q8M1QV,Q9  
M0QK,CARckB,U8MrOoB,S9MqOT,KAAL,KAAoB,CAVzB,U8M3N6B,S9M2NvB,yB8M3NgC,E9M2NhC,CA  
AN,CAUyB,MAApB,CAAN,CAqCIB,MAAK,Q8M1QV,I;K;sCAE5B,Y;MAAkC,OAAE,UAAF,qBAAU,S;K;IAE  
5C,gC;MAAA,oC;MACI,aAC+B,iBAAW,6BAAM,UAAjB,EAA4B,6BAAM,UAAIC,C;K;;IAFnC,4C;MAAA,2C;  
QAAA,0B;OAAA,oC;K;;IAYJ,qD;MA4CI,wC;MatCI,IAAI,gBAAJ,C;QAAwB,MAAA,gCAAyB,wBAAzB,C;MA  
CrC,IAAI,sCAAJ,C;QAA4B,MAAA,gCAAyB,yEAAzB,C;MAG7C,aAG0B,K;MAE1B,YAGyB,4BAA0B,KAA1B,  
EAAiC,YAAjC,EAA+C,IAA/C,C;MAEzB,YAGwB,I;K;0CAExB,Y;MAAiD,oCAAyB,UAAzB,EAAgC,SAAhC,E  
AAsC,SAAtC,C;K;yCAEjD,Y;MAMqC,OAAI,uBAAO,CAAX,G9Mf8B,a8MehB,U9MfkC,KAAIB,E8MeR,S9Mfs  
C,KAA9B,C8MehB,IAAd,G9Mf8B,a8MeE,U9MfgB,KAAIB,E8MeU,S9MfoB,KAA9B,C8MeE,I;K;wCAErE,iB;  
MAEY,UAAwB,M;MADhC,kDAA8B,kBAAa,KAAM,UAAAnB,KACtB,2CAAS,KAAM,MAAf,cAAwB,6CAAQ,  
KAAM,KAAd,QAAxB,KAA8C,kBAAQ,KAAM,KAAd,CADxB,CAA9B,C;K;0CAGJ,Y;MACI,OAAI,cAAJ,GAA  
e,EAAf,GAAwB,OAAM,M9MkND,CARckB,U8M7KX,U9M6KsB,KAAL,KAAoB,CAVzB,U8MnKD,U9MmKO,  
yB8MnKG,E9MmKH,CAAN,CAUyB,MAApB,CAAN,CAqCIB,MAAK,Q8MINJ,Q9MkND,CARckB,U8M7K0B,  
S9M6Kf,KAAL,KAAoB,CAVzB,U8MnKmC,S9MmK7B,yB8MnKsC,E9MmKtC,CAAN,CAUyB,MAApB,CAAN  
,CAqCIB,MAAK,Q8MINJ,IAAN,SAaQF,cAAU,6BAAU,EAAV,CAAyB,QAA9G,I;K;0CAE5B,Y;MAAk  
C,OAAI,uBAAO,CAAX,GAAgB,UAAF,qBAAU,SAAV,cAAqB,SAArB,WAAd,GAAgD,UAAF,2BAAgB,SAAh  
B,cAA6B,SAAD,aAA5B,W;K;IAEhF,sC;MAAA,0C;K;mEACI,sC;MAQ+F,4BAAiB,UAAjB,EAA6B,QAA7B,EA  
AuC,IAAvC,C;K;;IATnG,kD;MAAA,iD;QAAA,gC;OAAA,0C;K;;IAoBkC,qD;MAA0C,wB;MAC5E,sBAA2B,I;  
MAC3B,iBAAmC,kBAAO,CAA1C,G9MhDmE,a8MgDtB,K9MhDwC,KAAIB,E8MgDb,I9MhD2C,KAA9B,C8M  
gDtB,KAA7C,G9MhDmE,a8MgDH,K9MhDqB,KAAIB,E8MgDM,I9MhDwB,KAA9B,C8MgDH,K;MACHe,c9M  
0SsC,U8M1SnB,I9M0SmB,C;M8MzStC,cAAuB,cAAJ,GAAa,KAAb,GAAwB,mB;K;iDAE3C,Y;MAAkC,qB;K;m  
DAEIC,Y;MACI,YAAY,W;MACZ,IAAI,6BAAS,mBAAT,QAAJ,C;QACI,IAAI,CAAC,cAAL,C;UAAc,MAAA,6B  
;QAC3B,iBAAU,K;;QAEV,c9M/C+C,U8M+C/C,W9M/C0D,KAAK,K8M+CvD,W9M/CkE,KAAX,CAAhB,C;;M  
8MiDnD,OAAO,K;K;;wEC7Hf,yB;MAAA,8C;MAAA,uB;QAOI,OAAO,MAAM,CAAN,EAAS,CAAT,C;O;KAP  
X,C;wEAUA,yB;MAAA,8C;MAAA,uB;QAOI,OAAO,MAAM,CAAN,EAAS,CAAT,C;O;KAPX,C;wEAUA,yB;  
MAAA,8C;MAAA,uB;QAOI,OAAO,MAAM,CAAN,EAAS,CAAT,C;O;KAPX,C;wEAUA,yB;MAAA,8C;MAAA  
,uB;QAOI,OAAO,MAAM,CAAN,EAAS,CAAT,C;O;KAPX,C;oFC7BA,yB;MAAA,gD;MAAA,4B;QAM6C,OAA  
Q,ahO+RhB,cgO/RgB,C;O;KANrD,C;oGAQA,yB;M/GwCA,iB;M+GxCA,4B;QAMqD,O/GwCM,MAAO,OjH+O  
7B,ciH/O6B,C;O;K+G9CIE,C;sGAQA,yB;MAAA,kE;MAAA,4B;QAMsD,OAAQ,sBhO+QzB,cgO/QyB,C;O;KA  
N9D,C;8FAQA,yB;MAAA,0D;MhOwWA,6B;MgOxWA,4B;QAOmD,OhO2WZ,cgO3WoB,kBhOsQtB,cgOtQsB,  
ChO2WpB,C;O;KgOIXvC,C;4FASA,yB;MAAA,wD;MhO+VA,6B;MgO/VA,4B;QAOkD,OhOkWX,cgOIWmB,iB

hO6PrB, cgO7PqB, ChOkWnB, C;O; KgOzWvC, C; gFASA, yB; MAAA, 4C; MhOsVA, 6B; MgOtVA, sC; QAayD, OhOmVIB, cgOnV0B, WhO8O5B, cgO9O4B, EAAW, QAAX, ChOmV1B, C;O; KgOhWvC, C; kFAGBA, yB; MAAA, 8C; MhOsUA, 6B; MgOtUA, sC; QAa0D, OhOmUnB, cgOnU2B, YhO8N7B, cgO9N6B, EAAY, QAAZ, ChOmU3B, C;O; KgOhVvC, C; oFAGBA, yB; MAAA, gD; MAAA, 4B; QAM8C, OAAS, ahNgOhB, cgNhOgB, C;O; KANvD, C; oGAQA, yB; MAAA, gE; MAAA, 4B; QAMsD, OAAS, qBhNwNxB, cgNxNwB, C;O; KAN/D, C; sGAQA, yB; MAAA, kE; MAAA, 4B; QAMuD, OAAS, sBhNgNzB, cgNhNyB, C;O; KANhE, C; 8FAQA, yB; MAAA, 0D; MhN6SA, +B; MgN7SA, 4B; QAOqD, OhNgTX, egNhToB, kBhNuMvB, cgNvMuB, ChNgTpB, C;O; KgNvT1C, C; 4FASA, yB; MAAA, wD; MhNoSA, +B; MgNpSA, 4B; QAOoD, OhNuSV, egNvSmB, iBhN8LtB, cgN9LsB, ChNuSnB, C;O; KgN9S1C, C; +EASA, yB; MAAA, 4C; MhN2RA, +B; MgN3RA, sC; QAa2D, OhNwRjB, egNxR0B, WhN+K7B, cgN/K6B, EAAW, QAAX, ChNwR1B, C;O; KgNrS1C, C; iFAeA, yB; M/GgEA, 4C; MjG4MA, +B; MgN5QA, sC; QAa4D, OhNyQIB, eiGzMuB, WjGgG1B, ciGhG0B, EAAW, C+GhEK, Q/GgEL, IAAX, CjGyMvB, C;O; KgNtR1C, C; oFAeA, yB; MjOwJI, 6B; MiO1SJ, gD; MAKJA, 4B; QAM8C, OALJO, ahO+RhB, CDcE, cAAU, cAAL, GAAiB, GAAtB, CCdF, MgO/RgB, C;O; KA4IrD, C; oGAQA, yB; M/G1GA, iB; M+G0GA, 4B; QAMsD, O/G1GK, MAAO, OIHuM3B, c0N1Ge, GAAY, GxG7FA, CwG6Fb, GAA6C, EAA7C, I;O; KOOR, C; sGAQA, yB; MPbA, kE; MOaA, 4B; QAMuD, OPbkB, sB1NkGIC, c0NIGgB, GAAW, GAAO, C;O; KOOzE, C; 8FAQA, yB; MAAA, 0D; MjO+LA, 0B; MAAA, +B; MiO/LA, 4B; QAOqD, OjOmMZ, eAAW, OiOnMS, kBjOgGnB, cAAL, GAAiB, GiOhGO, CjOmMT, CAAX, C;O; KiO1MzC, C; 4FASA, yB; MAAA, wD; MjOsLA, 0B; MAAA, +B; MiOtLA, 4B; QAOoD, OjO0LX, eAAW, OiO1LQ, iBjOuFIB, cAAL, GAAiB, GiOvFM, CjO0LR, CAAX, C;O; KiOjMzC, C; gFAUA, yB; MAAA, 4C; MjOqJA, +B; MiOrJA, sC; QAa2D, OjOkJjB, eiOIJ0B, WjOmD7B, ciOnD6B, EAAW, QAAX, CjOkJ1B, C;O; KiO/J1C, C; kFAeA, yB; MAAA, 8C; MjOsIA, +B; MiOtIA, sC; QAa4D, OjOmIIB, eiOnI2B, YjOoC9B, ciOpC8B, EAAY, QAAZ, CjOmI3B, C;O; KiOhJ1C, C; oFAeA, yB; M/NgFI, 6B; M+N3SJ, gD; MA2NA, 4B; QAM+C, OA3NM, ahO+RhB, CCeE, cAAU, cAAL, GAAiB, KAAtB, CDfF, MgO/RgB, C;O; KAqNrD, C; oGAQA, yB; M/GnLA, iB; M+GmLA, 4B; QAMuD, O/GnLI, MAAO, OhHkNzB, cwN3CpC, GAAY, KxGvKiD, CwGuK9D, GAA+C, EAA/C, I;O; KOMJ, C; sGAQA, yB; MPZA, kE; MOYA, 4B; QAMwD, OPZoB, sBxNmCnC, cwNnCe, GAAW, KAAS, C;O; KOM5E, C; 8FAQA, yB; MAAA, 0D; M/NuHA, 4B; MAAA, iC; M+NvHA, 4B; QAOuD, O/N2HZ, gBAAY, Q+N3HQ, kB/NwBrB, cAAL, GAAiB, K+NxB, C/N2HR, CAAZ, C;O; K+NII3C, C; 4FASA, yB; MAAA, wD; M/N8GA, 4B; MAAA, iC; M+N9GA, 4B; QAOsD, O/NkHX, gBAAY, Q+NIHO, iB/NepB, cAAL, GAAiB, K+NfQ, C/NkHP, CAAZ, C;O; K+NzH3C, C; gFAUA, yB; MAAA, 4C; M/NyFA, iC; M+NzFA, sC; QAa6D, O/NsFhB, gB+NtF0B, W/NX9B, c+Nw8B, EAAW, QAAX, C/NsF1B, C;O; K+NnG7C, C; kFAeA, yB; MAAA, 8C; M/N0EA, iC; M+N1EA, sC; QAa8D, O/NuEjB, gB+NvE2B, Y/N1B/B, c+N0B+B, EAAY, QAAZ, C/NuE3B, C;O; K+NpF7C, C; ICtRA, qC; MAEI, SjOuIoD, ciOvI3C, CjOuI2C, EiOvIvC, CjOuIuC, C; MiOtIpD, SjOsIoD, ciOtl3C, CjOsI2C, EiOtlvC, CjOsIuC, C; MiOrIpD, OjOmDkE, YiOnDvD, EjOmDwE, KAAjB, EiOnDjD, EjOmD8E, KAA7B, CiOnDvD, KAAX, GjOkFsD, SiOIFjC, EjOkF2C, KAAK, GiOIF3C, EjOkFuD, KAAZ, IAaf, CiOIFtD, GjOqEqD, SAAU, CAaT, SiOIFpB, EjOkF8B, KAAK, GiOIF9B, EjOkF0C, KAAZ, IAaf, CAbs, MAAK, GiOrExB, CjOqEmC, KAAX, IAaf, C;K; liOIEzD, qC; MACI, SjnWIsD, eiNxI7C, CjNwI6C, EiNxIzC, CjNwIyC, C; MiNvItD, SjnNIsD, eiNvI7C, CjNuI6C, EiNvIzC, CjNuIyC, C; MiNtItD, OjNqDmE, aiNrDxD, EjNqD0E, KAAIB, EiNrDID, EjNqDgF, KA9B, CiNrDxD, KAAX, GjN+EwD, UiN/EnC, EjN+E8C, KAAK, UiN/E9C, EjN+E0D, KAAZ, CAAhB, CiN/ExD, GjNkEuD, UAAW, CAaV, UiN/EtB, EjN+EiC, KAAK, UiN/EjC, EjN+E6C, KAAZ, CAAhB, CAbu, MAAK, KiNIE3B, CjNkEsC, KAAX, CAAhB, C;K; liN/D3D, uD; MAmBI, WAAO, CAAP, C; QAD8E, OjOwBZ, YiOvBID, KjOuBmE, KAAjB, EiOvBzC, GjOuBsE, KAA7B, CiOvBID, KAD8D, GACHd, GADgD, GjOuDxB, SiOtDf, GjOsDyB, KAAK, GiOtDxB, mBAAiB, GAAjB, EAAsB, KAAtB, EjO2WV, SiO3WuC, IjO2WvC, CiO3WU, CjOsDoC, KAAZ, IAaf, C; aiOrDtD, WAAO, CAAP, C; QAF8E, OjOwBZ, YiOtBID, KjOsBmE, KAAjB, EiOtBzC, GjOsBsE, KAA7B, CiOtBID, KAF8D, GAEhD, GAFgD, GjO0CzB, SiOxCd, GjOwCwB, KAAK, GiOxCvB, mBAAiB, KAAjB, EAAwB, GAAxB, EjO0WV, SiO1WwC, CAAC, IAAD, IjO0WxC, CiO1WU, CjOwCkC, KAAX, IAaf, C; QiOvC7C, MAAa, gCAAYB, eAAzB, C;K; IAGzB, uD; MAmBI, sBAAO, CAAP, C; QADkF, OjNqF, aiNPNd, KjNOqE, KAAIB, EiNP1C, GjNOwE, KAA9B, CiNPNd, KADkE, GACpD, GADoD, GjNkC1B, UiNjCjB, GjNiC4B, KAAK, UiNjC3B, mBAAiB, GAAjB, EAAsB, KAAtB, EjNkWP, UiNIWoC, IjNkWP, CiNIWO, CjNiCuC, KAAZ, CAAhB, C; aiNhCx, sBAAO, CAAP, C; QAFkF, OjNqF, aiNNnD, KjNMqE, KAAIB, EiNN1C, GjNMwE, KAA9B, CiNNnD, KAFkE, GAEPD, GAFoD, GjNqB3B, UiNnBhB, GjNmB2B, KAAK, KiNnB1B, mBAAiB, KAAjB, EAAwB, GAAxB, EjNiWP, UiNjWsC, IAAD, ajNiWrC, CiNjWO, CjNmBqC, KAAX, CAAhB, C; QiNIB/C, MAAa, gCAAYB, eAAzB, C;K; IhOIDC, sB; MAEtB, 8B; MAFyD, gB; K; IAEzD, 4B; MAAA, gC; MAC

I,iBAGqC,WAAO,CAAP,C;MAErC,iBAGqC,WAAO,MAAP,C;MAErC,kBAGmC,C;MAEnC,iBAGkC,E;K;;;IAN  
BtC,wC;MAAA,uC;QAAA,sB;OAAA,gC;K;wGAsBA,iB;MAM0D,OAAa,0BA6OjC,SAAL,GAAiB,KA7OqB,EA  
AU,KF403C,KAAL,GAAiB,GE5OqB,C;K;oGAEvE,iB;MAOoE,OAAa,0BAoO3C,SAAL,GAAiB,KApO+B,EAA  
U,KAoOrD,KAAL,GAAiB,KApO+B,C;K;wGAEjF,yB;MA2PA,6B;MD5PA,8C;MCCA,wB;QAMyD,ODAS,YAA  
iB,CC8PhD,cAAU,SAAL,GAAiB,KAAtB,CD9PgD,MAAjB,ECAe,KDAc,KAA7B,C;O;KCNIE,C;wGAQA,yB;M  
A6PA,aAS6D,0B;MAT7D,+B;Me9PA,gD;MfCA,wB;QAM0D,OeAS,aAAkB,CfgQhD,eAAW,oBAAL,SAAK,CA  
AL,YAAN,CehQgD,MAAIB,EfAgB,KeAc,KAA9B,C;O;KfNnE,C;8FAQA,yB;MA2OA,6B;MA3OA,wB;QAEsD,  
ODMD,cAAU,CC4O5B,cAAU,SAAL,GAAiB,KAAtB,CD5O4B,MAAK,GAAW,CD2O5C,cEjPsC,KFiP5B,KAA  
L,GAAiB,GAAtB,CC3O4C,MAAX,IAAf,C;O;KCRrD,C;8FAGA,yB;MAwOA,6B;MAxOA,wB;QAEuD,ODGF,c  
AAU,CC4O5B,cAAU,SAAL,GAAiB,KAAtB,CD5O4B,MAAK,GAAW,CC4O5C,cA/OuC,KA+O7B,KAAL,GAAi  
B,KAAtB,CD5O4C,MAAX,IAAf,C;O;KCLrD,C;8FAGA,yB;MAqOA,6B;MArOA,wB;QAEqD,ODAA,cAAU,CC  
4O5B,cAAU,SAAL,GAAiB,KAAtB,CD5O4B,MAAK,GCAI,KDAO,KAAZ,IAAf,C;O;KCFrD,C;8FAGA,yB;MA  
4OA,aAS6D,0B;MAT7D,+B;MA5OA,wB;QAEuD,OeAA,eAAW,CfmP7B,eAAW,oBAAL,SAAK,CAAL,YAAN,  
CenP6B,MAAK,KfAI,KeAO,KAAZ,CAAhB,C;O;KfFvD,C;gGAIA,yB;MA8NA,6B;MA9NA,wB;QAEuD,ODMD  
,cAAU,CC+N7B,cAAU,SAAL,GAAiB,KAAtB,CD/N6B,MAAK,GAAZ,CD8N9C,cEpOwC,KFoO9B,KAAL,GA  
AiB,GAAtB,CC9N8C,MAAZ,IAAf,C;O;KCRtD,C;gGAGA,yB;MA2NA,6B;MA3NA,wB;QAEwD,ODGF,cAAU,  
CC+N7B,cAAU,SAAL,GAAiB,KAAtB,CD/N6B,MAAK,GAAZ,CC+N9C,cAlOyC,KAKO/B,KAAL,GAAiB,KAA  
tB,CD/N8C,MAAZ,IAAf,C;O;KCLtD,C;gGAGA,yB;MAwNA,6B;MAXNA,wB;QAEsD,ODAA,cAAU,CC+N7B,c  
AAU,SAAL,GAAiB,KAAtB,CD/N6B,MAAK,GCAK,KDAO,KAAZ,IAAf,C;O;KCFtD,C;gGAGA,yB;MA+NA,a  
AS6D,0B;MAT7D,+B;MA/NA,wB;QAEwD,OeAA,eAAW,CfsO9B,eAAW,oBAAL,SAAK,CAAL,YAAN,CetO8B  
,MAAK,UfAK,KeAO,KAAZ,CAAhB,C;O;KfFxD,C;gGAIA,yB;MAiNA,6B;MAjNA,wB;QAEuD,ODMD,cAAe,  
YAAL,CCKn7B,cAAU,SAAL,GAAiB,KAAtB,CDiN6B,MAAK,EAAZ,CDiN9C,cEvNwC,KFuN9B,KAAL,GAAi  
B,GAAtB,CCjN8C,MAAZ,CAAf,C;O;KCRtD,C;gGAGA,yB;MA8MA,6B;MA9MA,wB;QAEwD,ODGF,cAAe,Y  
AAL,CCKn7B,cAAU,SAAL,GAAiB,KAAtB,CDiN6B,MAAK,EAAZ,CCKn9C,cArNyC,KAqN/B,KAAL,GAAiB,  
KAAtB,CDiN8C,MAAZ,CAAf,C;O;KCLtD,C;gGAGA,yB;MA2MA,6B;MA3MA,wB;QAEsD,ODAA,cAAe,YAA  
L,CCKn7B,cAAU,SAAL,GAAiB,KAAtB,CDiN6B,MAAK,ECAK,KDAO,KAAZ,CAAf,C;O;KCFtD,C;gGAGA,y  
B;MAkNA,aAS6D,0B;MAT7D,+B;MAINA,wB;QAEwD,OeAA,eAAW,CfyN9B,eAAW,oBAAL,SAAK,CAAL,Y  
AAN,CezN8B,MAAK,UfAK,KeAO,KAAZ,CAAhB,C;O;KfFxD,C;4FAIA,yB;MAoMA,6B;MD9LA,4C;MCNA,w  
B;QAEqD,ODMD,WCqMjB,cAAU,SAAL,GAAiB,KAAtB,CDrMiB,EDoMjB,cE1MoC,KF0M1B,KAAL,GAAiB,  
GAAtB,CCpMiB,C;O;KCRpD,C;4FAGA,yB;MAiMA,6B;MD9LA,4C;MCHA,wB;QAEsD,ODGF,WCqMjB,cAA  
U,SAAL,GAAiB,KAAtB,CDrMiB,ECqMjB,cAxMqC,KAwM3B,KAAL,GAAiB,KAAtB,CDrMiB,C;O;KCLpD,C;  
4FAGA,yB;MA8LA,6B;MD9LA,4C;MCAA,wB;QAEoD,ODAA,WCqMjB,cAAU,SAAL,GAAiB,KAAtB,CDrMi  
B,ECAkB,KDAIB,C;O;KCFpD,C;4FAGA,yB;MAqMA,aAS6D,0B;MAT7D,+B;MerMA,8C;MfAA,wB;QAEsD,O  
eAA,Yf4MjB,eAAW,oBAAL,SAAK,CAAL,YAAN,Ce5MiB,EfAmB,KeAnB,C;O;KfFtD,C;4FAIA,yB;MAuLA,6  
B;MDzKA,kD;MCdA,wB;QAMqD,ODcD,cC4KjB,cAAU,SAAL,GAAiB,KAAtB,CD5KiB,ED2KjB,cEzLoC,KFy  
L1B,KAAL,GAAiB,GAAtB,CC3KiB,C;O;KCPbD,C;4FAOA,yB;MAGLA,6B;MDzKA,kD;MCPA,wB;QAMsD,  
ODOF,cC4KjB,cAAU,SAAL,GAAiB,KAAtB,CD5KiB,EC4KjB,cAnLqC,KAmL3B,KAAL,GAAiB,KAAtB,CD5K  
iB,C;O;KCbD,C;4FAOA,yB;MAyKA,6B;MDzKA,kD;MCAA,wB;QAMoD,ODAA,cC4KjB,cAAU,SAAL,GAAi  
B,KAAtB,CD5KiB,ECAkB,KDAIB,C;O;KCNpD,C;4FAOA,yB;MA4KA,aAS6D,0B;MAT7D,+B;Me5KA,oD;Mf  
AA,wB;QAMsD,OeAA,ef+KjB,eAAW,oBAAL,SAAK,CAAL,YAAN,Ce/KiB,EfAmB,KeAnB,C;O;KfNtD,C;sGA  
QA,yB;MA0JA,6B;MD9LA,4C;MCoCA,wB;QAMiD,ODxCG,WCqMjB,cAAU,SAAL,GAAiB,KAAtB,CDrMiB,E  
DoMjB,cE5JqC,KF4J3B,KAAL,GAAiB,GAAtB,CCpMiB,C;O;KCKpD,C;sGAOA,yB;MAmJA,6B;MD9LA,4C;  
MC2CA,wB;QAMkD,OD/CE,WCqMjB,cAAU,SAAL,GAAiB,KAAtB,CDrMiB,ECqMjB,cAtJsC,KAsJ5B,KAAL,  
GAAiB,KAAtB,CDrMiB,C;O;KCyCpD,C;sGAOA,yB;MA4IA,6B;MD9LA,4C;MCKDA,wB;QAMgD,ODtDI,WCq  
MjB,cAAU,SAAL,GAAiB,KAAtB,CDrMiB,ECsDmB,KDtDnB,C;O;KCGDpD,C;sGAOA,yB;MA+IA,aAS6D,0B;  
MAT7D,+B;MerMA,8C;MfsDA,wB;QAMkD,Oe1DI,Yf4MjB,eAAW,oBAAL,SAAK,CAAL,YAAN,Ce5MiB,Ef0  
DoB,Ke1DpB,C;O;KfoDtD,C;4FAQA,yB;MA6HA,6B;MDzKA,kD;MDuOJ,0B;MAAA,+B;ME3LI,wB;QAQ6C,O  
F8LR,eAAW,OC5OI,cC4KjB,cAAU,SAAL,GAAiB,KAAtB,CD5KiB,ED2KjB,cE7H4B,KF6HIB,KAAL,GAAiB,

GAAAtB,CC3KiB,CAKlf,KD0DW,CAAX,C;O;KEtMrC,C;4FASA,yB;MAoHA,6B;MDzKA,kD;MCwOJ,4B;MAA A,iC;MAnLI,wB;QAQ+C,OAsLR,gBAAY,QD7OC,cC4KjB,cAAU,SAAL,GAAiB,KAAAtB,CD5KiB,EC4KjB,cAr H8B,KAqHpB,KAAL,GAAiB,KAAAtB,CD5KiB,CA4Lb,KCiDY,CAAZ,C;O;KA9LvC,C;4FASA,yB;MA2GA,6B; MDzKA,kD;MC8DA,wB;QAQ2C,ODhES,cC4KjB,cAAU,SAAL,GAAiB,KAAAtB,CD5KiB,ECgES,KDhET,C;O;K CwDpD,C;4FASA,yB;MA4GA,aAS6D,0B;MAT7D,+B;Me5KA,oD;MfgEA,wB;QAQ6C,OelES,ef+KjB,eAAW,o BAAL,SAAK,CAAL,YAAN,Ce/KiB,EfkEU,KeLEV,C;O;Kf0DtD,C;4EAUUA,yB;MAAA,4B;MAAA,iC;MAAA,m B;QAM2C,uBAAY,QAAL,SAAK,KAAZ,C;O;KAN3C,C;4EAQA,yB;MAAA,4B;MAAA,iC;MAAA,mB;QAM2C ,uBAAY,QAAL,SAAK,KAAZ,C;O;KAN3C,C;oGAQA,yB;MAAA,8C;MAwEA,6B;MAxEA,wB;QAE+D,0BA+E 5B,cAAU,SAAL,GAAiB,KAAAtB,CA/E4B,EA+E5B,cA/EqD,KA+E3C,KAAL,GAAiB,KAAAtB,CA/E4B,C;O;KAF/ D,C;4FAIA,yB;MAAA,iC;M0LnNJ,4B;M1LmNI,wB;QAEqD,uB0LlNiC,Q1LkN1B,IAAK,K0LINX,G1LkNoB,K AAM,K0LINM,C1LkNjC,C;O;KAFrD,C;0FAGA,yB;MAAA,iC;M0LjNJ,4B;M1LiNI,wB;QAEoD,uB0LhNgC,Q1 LgNzB,IAAK,K0LhNX,G1LgNmB,KAAM,K0LhNM,C1LgNhC,C;O;KAFpD,C;4FAGA,yB;MAAA,iC;M0L/MJ,4 B;M1L+MI,wB;QAEqD,uB0L9MiC,Q1L8M1B,IAAK,K0L9MX,G1L8MoB,KAAM,K0L9MM,C1L8MjC,C;O;KA FrD,C;4EAGA,yB;MAAA,iC;M0L7MJ,4B;M1L6MI,mB;QAEkC,uB0L5MsB,QAAP,C1L4MR,S0L5Me,C1L4Mt B,C;O;KAFIC,C;kFAIA,yB;MAAA,0B;MAAA,mB;QAUmC,OAAK,OAAL,SAAK,C;O;KAVxC,C;oFAWA,Y;M ASqC,gB;K;gFACrC,Y;MASiC,OAAK,SAAL,GAAiB,K;K;kFACID,yB;MAAA,aASqD,0B;MATrD,mB;QASmC, OAAK,oBAAL,SAAK,CAAL,Y;O;KATnC,C;oFAWA,yB;MF+DJ,0B;MAAA,+B;ME/DI,mB;QASqC,OFiEE,eA AW,OEjEb,SfIEa,CAAX,C;O;KE1EvC,C;sFAUA,Y;MAEuC,W;K;kFACvC,yB;MAAA,6B;MAAA,mB;QASmC, qBAAU,SAAL,GAAiB,KAAAtB,C;O;KATnC,C;oFAUA,yB;MAAA,aAS6D,0B;MAT7D,+B;MAAA,mB;QASqC,s BAAW,oBAAL,SAAK,CAAL,YAAN,C;O;KATrC,C;oFAWA,Y;MAMqC,OApDC,SAAL,GAAiB,K;K;sFAqDID, Y;MAMuC,OA3DD,SAAL,GAAiB,K;K;gCA6DID,Y;MAAyC,OAAQ,CA7DX,SAAL,GAAiB,KA6DD,Y;K;,,,gC A3UrD,Y;MAAA,c;MAG6D,qD;MAH7D,a;K;8BAAA,iB;MAAA,2IAG6D,oCAH7D,G;K;0EA+UA,yB;MAAA,i C;MAAA,4B;QAW4C,uBAAY,SAAZ,C;O;KAX5C,C;4EAYA,yB;MAAA,iC;MAAA,4B;QAU6C,uBAAO,SAAP, C;O;KAV7C,C;4EAWA,yB;MAAA,4B;MAAA,iC;MAAA,4B;QAW2C,uBAAY,QAAL,SAAK,CAAZ,C;O;KAX3 C,C;4EAYA,yB;MAAA,4B;MAAA,iC;MAAA,4B;QAW4C,uBAAY,QAAL,SAAK,SAAZ,C;O;KAX5C,C;liC/W A,8B;MACqB,sB;K;wCAKjB,iB;MAM8C,OjCsVL,WiCtVK,aAAQ,KAAR,CjCsVL,C;K;wCiCpVzC,wB;MAOI,a AAQ,KAAR,IAAiB,KjC4OgB,K;K;mFiCxOP,Y;MAAQ,OAAA,YAAQ,O;K;qCAE9C,Y;MAC+E,gCAAS,YAAT, C;K;IAGzD,qC;MAAkC,yB;MAAjC,oB;MACnB,eAAoB,C;K;6CACpB,Y;MAAyB,sBAAQ,YAAM,O;K;gDACv C,Y;MAA0D,Q;MAA9B,IAAI,eAAQ,YAAM,OAAIB,C;QAAA,OjCgUS,WiChUe,aAAM,mBAAN,EAAM,2BAA N,OjCgUf,C;Q;QiChU8C,MAAM,2BAAuB,YAAM,WAA7B,C;K;,,,2CAG7F,mB;MAIS,Q;MAAL,IAAI,eAAC,0EA AD,SAAJ,C;QAAkC,OAAO,K;MAEzC,OAAe,WAAR,YAAQ,EAAS,OjCsNS,KiCtNIB,C;K;gDAGnB,oB;MACY ,Q;MAA2B,gBAA3B,gE;MAA2B,c;QjB0nDvB,U;QADhB,IAAI,wCAAsB,mBAA1B,C;UAAqC,aAAO,I;UAAP,e ;SACrB,6B;QAAhB,OAAgB,gBAAhB,C;UAAgB,2B;UiB1nD6B,2BjB0nDR,OiB1nDQ,S;UAAA,W;YAAwB,oB AAR,YAAQ,EjB0nDhC,OhBx6CA,KiCINgC,C;WjB0nD/C,IAAI,OAAJ,C;YAAyB,aAAO,K;YAAP,e;QAC/c,aA AO,I;,,,MiB3nDH,iB;K;oCAGJ,Y;MAAkC,OAAA,IAAK,QAAQ,OAAb,KAAqB,C;K;IA/CvD,uC;MAAA,qD;MA CgC,wBAAK,eAAW,IAAX,CAAL,C;MADhC,Y;K;,,,qCAPJ,Y;MAAA,OAKqB,sDALrB,M;K;qCAAA,Y;MAAA ,c;MAKqB,wD;MALrB,a;K;mCAAA,iB;MAAA,2IAKqB,0CALrB,G;K;kFAyDA,yB;MAAA,2C;MAWwC,0C;QA AA,wB;UAAW,OAAA,aAAK,KAAL,CjCiMV,K;S;O;MiC5MzC,6B;QAWI,OAAO,qBAAY,gCAAW,IAAX,GAA iB,wBAAjB,CAAZ,C;O;KAXX,C;oFAcA,oB;MAGwE,e;K;Igm5ExE,sC;MAQ2D,OAAa,WAAb,SnOwQjB,KAA L,GAAiB,GmOxQkB,EAAS,KAAT,C;K;IAExE,sC;MAQ4D,OAAa,WAAb,SjO+PIB,KAAL,GAAiB,KiO/PmB,E AAS,KAAT,C;K;IAGzE,sC;MAQ0D,OAAc,WIOiR5B,oBkOjRc,SIOiRnB,KAAK,CAAL,iBkOjRiC,EAAS,KAAT ,C;K;IAExE,sC;MAOgD,uBAAc,SlNyQvB,KkNzQS,EAA6B,WAAW,KAAX,CAA7B,C;K;IAGhD,8B;MAMqC,Q ;MAAA,0DAAmB,kBAAkB,SAAlB,C;K;IAExD,qC;MAO+C,Q;MAAA,0CAAc,KAAAd,oBAAwB,kBAAkB,SAAl B,C;K;IAGvE,+B;MAMuC,Q;MAAA,2DAAoB,kBAAkB,SAAlB,C;K;IAE3D,sC;MAOiD,Q;MAAA,2CAAE,KA Af,oBAAyB,kBAAkB,SAAlB,C;K;IAE1E,6B;MAMmC,Q;MAAA,yDAAkB,kBAAkB,SAAlB,C;K;IAErD,oC;MA O6C,Q;MAAA,yCAAA,KAAb,oBAAuB,kBAAkB,SAAlB,C;K;IAEpE,8B;MAMqC,Q;MAAA,0DAAmB,kBAAkB ,SAAlB,C;K;IAExD,qC;MAO+C,Q;MAAA,0CAAc,KAAAd,oBAAwB,kBAAkB,SAAlB,C;K;IAMvE,kC;MAM4C, kCAAsB,EAAtB,C;K;IAE5C,2C;MASmB,Q;MAAA,sBAAL,SAAK,EAAa,KAAb,C;MAAL,iB;QAA4B,OAAO,I;

OAA7C,UAAU,I;MACV,IIO/EkE,YkO+E9D,GIO/E+E,KAAjB,EAA6B,CD6P5D,SmO9KzB,6BAAM,UnO8K6B,KAAL,GAAiB,GAAtB,CC7P4D,MAA7B,CkO+E9D,IAAJ,C;QAA2B,OAAO,I;MACIC,OnO8OqC,UAAW,OmO9OzC,GIOoL8B,KD0Dw,CAAX,C;K;ImO3OzC,mC;MAM8C,mCAAuB,EAAvB,C;K;IAE9C,4C;MASmB,Q;MAA A,sBAAL,SAAK,EAAa,KAAb,C;MAAL,iB;QAA4B,OAAO,I;OAA7C,UAAU,I;MACV,IIOrGkE,YkOqG9D,GIOrG+E,KAAjB,EAA6B,CC8P5D,SIOzJzB,8BAAO,UjOyJ4B,KAAL,GAAiB,KAAtB,CD9P4D,MAA7B,CkOqG9D,IAAJ,C;QAA4B,OAAO,I;MACnC,OjOyNuC,WAAy,QiOzN5C,GIOwKgC,KCiDY,CAAZ,C;K;IiOtN3C,iC;MAM0C,iCAAqB,EAArB,C;K;IAE1C,0C;MASI,WAAW,KAAX,C;MAEA,aAAa,SAAK,O;MACIB,IAAI,WAAU,CAAd,C;QAAiB,OAAO,I;MAExB,YAAkB,4BAAK,U;MACvB,S;MAEA,gBAAGB,qBAAK,CAAL,C;MACHB,IAAI,YAAy,EAHb,C;QACI,IAAI,WAAU,CAAV,IAAe,cAAa,EAHc,C;UAAqC,OAAO,I;QAC5C,QAAQ,C;;QAER,QAAQ,C;;MAGZ,uBAAuB,mB;MAEvB,qBAAqB,gB;MACrB,alOuMmC,SkOvMtB,KIOuMsB,C;MkOfMnC,aAAa,W;MACb,aAAU,KAAY,MAASB,MAAtB,M;QACI,YAAy,QAAQ,qBAAK,CAAL,CAAR,EAAiB,KAAjB,C;QAEZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,IIONJ8D,YkOmJ1D,MIONJ2E,KAAjB,EkOmJjD,clOnJ8E,KAA7B,CkOmJ1D,IAAJ,C;UACI,IAAI,+CAAkB,gBAAIB,QAAJ,C;YACI,iBIO5FwC,WkO4FvB,KIO5FuB,EkO4Ff,MIO5Fe,C;YkO8FxC,IIOvJsD,YkOuJlD,MIOvJmE,KAAjB,EkOuJzC,clOvJsE,KAA7B,CkOuJlD,IAAJ,C;cACI,OAAO,I;;YAGX,OAAO,I;;SAIf,SIONHkD,SAAE,YkOmHjE,MIONH4D,KAAK,EkOmHvD,MIONHmE,KAAZ,CAAF,C;QkOqHID,mBAAmB,M;QACnB,SIOhJiD,SkOgJjD,MIOhJ2D,KAAK,GAAW,CAkU5C,SkOILrB,KIOkLqB,CAIU4C,MAAX,IAAf,C;QkOiJjD,IIONK8D,YkOmK1D,MIONK2E,KAAjB,EkOmKjD,YIONK8E,KAA7B,CkOmK1D,IAAJ,C;UAA2B,OAAO,I;;MAGtC,OAAO,M;K;IAGX,kC;MAM4C,kCAASB,EAAtB,C;K;IAE5C,2C;MASI,WAAW,KAAX,C;MAEA,aAAa,SAAK,O;MACIB,IAAI,WAAU,CAAd,C;QAAiB,OAAO,I;MAExB,YAAmB,6BAAM,U;MACzB,S;MAEA,gBAAGB,qBAAK,CAAL,C;MACHB,IAAI,YAAy,EAHb,C;QACI,IAAI,WAAU,CAAV,IAAe,cAAa,EAHc,C;UAAqC,OAAO,I;QAC5C,QAAQ,C;;QAER,QAAQ,C;;MAIZ,uBAAuB,gD;MAEvB,qBAAqB,gB;MACrB,alN0IqC,UAAW,oBkN1InC,KIN0ImC,CAAX,C;MkNzIrC,aAAa,2B;MACb,aAAU,KAAY,MAASB,MAAtB,M;QACI,YAAy,QAAQ,qBAAK,CAAL,CAAR,EAAiB,KAAjB,C;QAEZ,IAAI,QAAQ,CAAZ,C;UAAe,OAAO,I;QACtB,IIN5M+D,akN4M3D,MIN5M6E,KAAIB,EkN4MID,clN5MgF,KAA9B,CkN4M3D,IAAJ,C;UACI,IAAI,+CAAkB,gBAAIB,QAAJ,C;YACI,iBIN1JOC,YkN0JzB,KIN1JyB,EkN0JjB,MIN1JiB,C;YkN4J1C,IINhNuD,akNgNnD,MINhNqE,KAAIB,EkNgN1C,clNhNwE,KAA9B,CkNgNnD,IAAJ,C;cACI,OAAO,I;;YAGX,OAAO,I;;SAIf,SINjLoD,UkNiLpD,MINjL+D,KAAK,UkNiL1D,MINjLsE,KAAZ,CAAhB,C;QkNmLpD,mBAAmB,M;QACnB,SIN9MmD,UkN8MnD,MIN9M8D,KAAK,KAAW,ChBsQ7C,UAAW,oBAAL,CAYDR,SkOjHrB,KIOiHqB,CAZDQ,MAAK,CAAL,iBAAN,CgBtQ6C,MAAX,CAAhB,C;QkN+MnD,IIN5N+D,akN4N3D,MIN5N6E,KAAIB,EkN4NID,YIN5NgF,KAA9B,CkN4N3D,IAAJ,C;UAA2B,OAAO,I;;MAGtC,OAAO,M;K;IIN9RX,6B;MACKD,OAAUB,0BAAtB,KAAO,WAAe,EAAU,KAAO,WAAjB,C;K;IACzE,8B;MACqD,OAAC,gCAAuB,iBAAU,gCAAV,C;K;IAE7E,4B;MACoD,ORiZZ,SAvGI,oBQ1SS,ER0Sd,KAAK,CAAL,iBQ1Sc,KR0ST,oBQ1SuB,ER0S5B,KAAK,CAAL,iBQ1Sc,CRiZH,QAAV,C;K;IQhZxC,+B;MACuD,OR+Yf,SAvGI,oBQxSY,ERwSjB,KAAK,CAAL,iBQxSiB,QRwSZ,oBQxS0B,ERwS/B,KAAK,CAAL,iBQxSiB,CR+YN,QAAV,C;K;IQ1YxC,6B;MAEI,eAAe,EQkSoB,K;MRjSnC,cAAc,EQiSqB,K;MRhSnC,IAAI,qBAAU,CAAd,C;QACI,OQ6C+D,aR7CpD,EQ6CsE,KAAIB,ER7C/C,EQ6C6E,KAA9B,CR7CpD,IAAJ,GAAa,aAAb,GAA2B,a;OAItC,IAAI,uBAAy,CAAhB,C;QACI,OAAO,UAAM,aAAW,OAAX,CAAN,C;OAIx,eAAiB,4BAAC,CAAd,CAAD,KAAoB,OAAPB,CAAD,WAAkC,CAAIC,C;MACf,UAAU,kBAAW,kBAAW,OAAX,CAAX,C;MACV,OAAO,UAAM,iCQkCsD,aAAkB,CRICzD,UAAM,GAAN,CQkCyD,MAAIB,EAA8B,CRICvD,UAAM,OAAN,CQkCuD,MAA9B,CRICvC,KAAJ,GAAC,CAAIC,GAAYC,CAAPD,EAAN,C;K;IAIX,gC;MAKe,Q;MAHX,eAAe,EQ8QoB,K;MR7QnC,cAAc,EQ6QqB,K;MR5QnC,IAAI,qBAAU,CAAAd,C;QACW,IQyBwD,aRzBpD,EQyBsE,KAAIB,ERzB/C,EQyB6E,KAA9B,CRzBpD,IAAJ,C;UACH,S;;UAEA,OQgDgD,URhDhD,EQgD2D,KAAK,URhD3D,EQgDuE,KAAZ,CAAhB,C;;QRnDpD,W;OAQJ,IAAI,uBAAy,CAAhB,C;QACI,OAAO,UAAM,gBAAW,OAAX,CAAN,C;OAIx,eAAiB,4BAAC,CAAd,CAAD,KAAoB,OAAPB,CAAD,WAAkC,CAAIC,C;MACf,UAAU,kBAAW,kBAAW,OAAX,CAAX,C;MACV,OAAO,UAAM,aQUsD,aAAkB,CRV9D,UAAM,GAAN,CQU8D,MAAIB,EAA8B,CRV5D,UAAM,OAAN,CQU4D,MAA9B,CRV5C,KAAJ,GAAkC,OAAC,KAAN,CAAN,C;K;IAGX,yB;MAEI,IAAE,QAAF,CAAE,CAAF,C;QADyC,OAC5B,W;;QACb,SRwSuC,aQxSIC,4BAAK,URwS0C,KAAb,CQxSvC,C;UAFyC,OAEP,4BAAK,U;;UACvC,SRuSuC,aQvSIC,4BAAK,URuS0C,KAAb,CQvSvC,C;YAHyC,OAGP,4BAAK,U;eACvC,SAAK,UAAL,C;YAJyC,ORkVN,SQ9UX,YAAF,



## 1.27.1 Available under license :

### LICENSE ISSUES

=====

The OpenSSL toolkit stays under a double license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit.

See below for the actual license texts.

#### OpenSSL License

-----

```
/* =====  
* Copyright (c) 1998-2019 The OpenSSL Project. All rights reserved.  
*  
* Redistribution and use in source and binary forms, with or without  
* modification, are permitted provided that the following conditions  
* are met:  
*  
* 1. Redistributions of source code must retain the above copyright  
* notice, this list of conditions and the following disclaimer.  
*  
* 2. Redistributions in binary form must reproduce the above copyright  
* notice, this list of conditions and the following disclaimer in  
* the documentation and/or other materials provided with the  
* distribution.  
*  
* 3. All advertising materials mentioning features or use of this  
* software must display the following acknowledgment:  
* "This product includes software developed by the OpenSSL Project  
* for use in the OpenSSL Toolkit. (http://www.openssl.org/)"  
*  
* 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to  
* endorse or promote products derived from this software without  
* prior written permission. For written permission, please contact  
* openssl-core@openssl.org.  
*  
* 5. Products derived from this software may not be called "OpenSSL"  
* nor may "OpenSSL" appear in their names without prior written  
* permission of the OpenSSL Project.  
*  
* 6. Redistributions of any form whatsoever must retain the following  
* acknowledgment:  
* "This product includes software developed by the OpenSSL Project  
* for use in the OpenSSL Toolkit (http://www.openssl.org/)"  
*  
* THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY  
* EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
```

\* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR  
\* PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR  
\* ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,  
\* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT  
\* NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;  
\* LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)  
\* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,  
\* STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)  
\* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED  
\* OF THE POSSIBILITY OF SUCH DAMAGE.

\* =====

\*

\* This product includes cryptographic software written by Eric Young  
\* (eay@cryptsoft.com). This product includes software written by Tim  
\* Hudson (tjh@cryptsoft.com).

\*

\*/

#### Original SSLeay License

-----

/\* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)

\* All rights reserved.

\*

\* This package is an SSL implementation written  
\* by Eric Young (eay@cryptsoft.com).

\* The implementation was written so as to conform with Netscapes SSL.

\*

\* This library is free for commercial and non-commercial use as long as  
\* the following conditions are aheared to. The following conditions  
\* apply to all code found in this distribution, be it the RC4, RSA,  
\* lhash, DES, etc., code; not just the SSL code. The SSL documentation  
\* included with this distribution is covered by the same copyright terms  
\* except that the holder is Tim Hudson (tjh@cryptsoft.com).

\*

\* Copyright remains Eric Young's, and as such any Copyright notices in  
\* the code are not to be removed.

\* If this package is used in a product, Eric Young should be given attribution  
\* as the author of the parts of the library used.

\* This can be in the form of a textual message at program startup or  
\* in documentation (online or textual) provided with the package.

\*

\* Redistribution and use in source and binary forms, with or without  
\* modification, are permitted provided that the following conditions  
\* are met:

\* 1. Redistributions of source code must retain the copyright

\* notice, this list of conditions and the following disclaimer.

\* 2. Redistributions in binary form must reproduce the above copyright

- \* notice, this list of conditions and the following disclaimer in the
- \* documentation and/or other materials provided with the distribution.
- \* 3. All advertising materials mentioning features or use of this software
- \* must display the following acknowledgement:
- \* "This product includes cryptographic software written by
- \* Eric Young (eay@cryptsoft.com)"
- \* The word 'cryptographic' can be left out if the routines from the library
- \* being used are not cryptographic related :-).
- \* 4. If you include any Windows specific code (or a derivative thereof) from
- \* the apps directory (application code) you must include an acknowledgement:
- \* "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"
- \*
- \* THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND
- \* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
- \* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
- \* ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
- \* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
- \* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
- \* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
- \* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
- \* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
- \* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
- \* SUCH DAMAGE.
- \*
- \* The licence and distribution terms for any publically available version or
- \* derivative of this code cannot be changed. i.e. this code cannot simply be
- \* copied and put under another distribution licence
- \* [including the GNU Public Licence.]
- \*/

#### GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.

59 Temple Place - Suite 330, Boston, MA

02111-1307, USA.

Everyone is permitted to copy and distribute verbatim copies  
of this license document, but changing it is not allowed.

#### Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to

your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

## GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law:

that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If

identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the

operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under

any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

#### END OF TERMS AND CONDITIONS

#### Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>  
Copyright (C) 19yy <name of author>
```

```
This program is free software; you can redistribute it and/or modify  
it under the terms of the GNU General Public License as published by  
the Free Software Foundation; either version 2 of the License, or  
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,  
but WITHOUT ANY WARRANTY; without even the implied warranty of  
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the  
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License
```

along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands ``show w'` and ``show c'` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than ``show w'` and ``show c'`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.
```

```
<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

The "Artistic License"

Preamble

The intent of this document is to state the conditions under which a Package may be copied, such that the Copyright Holder maintains some semblance of artistic control over the development of the package, while giving the users of the package the right to use and distribute the Package in a more-or-less customary fashion, plus the right to make reasonable modifications.

Definitions:

"Package" refers to the collection of files distributed by the Copyright Holder, and derivatives of that collection of files created through textual modification.

"Standard Version" refers to such a Package if it has not been modified, or has been modified in accordance with the wishes of the Copyright Holder as specified below.

"Copyright Holder" is whoever is named in the copyright or copyrights for the package.

"You" is you, if you're thinking about copying or distributing this Package.

"Reasonable copying fee" is whatever you can justify on the basis of media cost, duplication charges, time of people involved, and so on. (You will not be required to justify it to the Copyright Holder, but only to the computing community at large as a market that must bear the fee.)

"Freely Available" means that no fee is charged for the item itself, though there may be fees involved in handling the item. It also means that recipients of the item may redistribute it under the same conditions they received it.

1. You may make and give away verbatim copies of the source form of the Standard Version of this Package without restriction, provided that you duplicate all of the original copyright notices and associated disclaimers.
2. You may apply bug fixes, portability fixes and other modifications derived from the Public Domain or from the Copyright Holder. A Package modified in such a way shall still be considered the Standard Version.
3. You may otherwise modify your copy of this Package in any way, provided that you insert a prominent notice in each changed file stating how and when you changed that file, and provided that you do at least ONE of the following:
  - a) place your modifications in the Public Domain or otherwise make them Freely Available, such as by posting said modifications to Usenet or an equivalent medium, or placing the modifications on a major archive site such as uunet.uu.net, or by allowing the Copyright Holder to include your modifications in the Standard Version of the Package.
  - b) use the modified Package only within your corporation or organization.

c) rename any non-standard executables so the names do not conflict with standard executables, which must also be provided, and provide a separate manual page for each non-standard executable that clearly documents how it differs from the Standard Version.

d) make other distribution arrangements with the Copyright Holder.

4. You may distribute the programs of this Package in object code or executable form, provided that you do at least ONE of the following:

a) distribute a Standard Version of the executables and library files, together with instructions (in the manual page or equivalent) on where to get the Standard Version.

b) accompany the distribution with the machine-readable source of the Package with your modifications.

c) give non-standard executables non-standard names, and clearly document the differences in manual pages (or equivalent), together with instructions on where to get the Standard Version.

d) make other distribution arrangements with the Copyright Holder.

5. You may charge a reasonable copying fee for any distribution of this Package. You may charge any fee you choose for support of this Package. You may not charge a fee for this Package itself. However, you may distribute this Package in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution provided that you do not advertise this Package as a product of your own. You may embed this Package's interpreter within an executable of yours (by linking); this shall be construed as a mere form of aggregation, provided that the complete Standard Version of the interpreter is so embedded.

6. The scripts and library files supplied as input to or produced as output from the programs of this Package do not automatically fall under the copyright of this Package, but belong to whoever generated them, and may be sold commercially, and may be aggregated with this Package. If such scripts or library files are aggregated with this Package via the so-called "undump" or "unexec" methods of producing a binary executable image, then distribution of such an image shall neither be construed as a distribution of this Package nor shall it fall under the restrictions of Paragraphs 3 and 4, provided that you do not represent such an executable image as a Standard Version of this Package.

7. C subroutines (or comparably compiled subroutines in other languages) supplied by you and linked into this Package in order to

emulate subroutines and variables of the language defined by this Package shall not be considered part of this Package, but are the equivalent of input as in Paragraph 6, provided these subroutines do not change the language in any way that would cause it to fail the regression tests for the language.

8. Aggregation of this Package with a commercial distribution is always permitted provided that the use of this Package is embedded; that is, when no overt attempt is made to make this Package's interfaces visible to the end user of the commercial distribution. Such use shall not be construed as a distribution of this Package.

9. The name of the Copyright Holder may not be used to endorse or promote products derived from this software without specific prior written permission.

10. THIS PACKAGE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The End

## 1.28 util-linux 2.31.1

### 1.28.1 Available under license :

/\*

- \* Copyright (c) 1989 The Regents of the University of California.
- \* All rights reserved.
- \*
- \* Redistribution and use in source and binary forms, with or without
- \* modification, are permitted provided that the following conditions
- \* are met:
- \* 1. Redistributions of source code must retain the above copyright
- \* notice, this list of conditions and the following disclaimer.
- \* 2. Redistributions in binary form must reproduce the above copyright
- \* notice, this list of conditions and the following disclaimer in the
- \* documentation and/or other materials provided with the distribution.
- \* 3. All advertising materials mentioning features or use of this software
- \* must display the following acknowledgement:
- \* This product includes software developed by the University of
- \* California, Berkeley and its contributors.
- \* 4. Neither the name of the University nor the names of its contributors
- \* may be used to endorse or promote products derived from this software
- \* without specific prior written permission.
- \*
- \* THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS" AND
- \* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
- \* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE

\* ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE  
\* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL  
\* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS  
\* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)  
\* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT  
\* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY  
\* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF  
\* SUCH DAMAGE.

\*/

This library is free software; you can redistribute it and/or  
modify it under the terms of the GNU Lesser General Public  
License as published by the Free Software Foundation; either  
version 2.1 of the License, or (at your option) any later  
version.

The complete text of the license is available in the  
../Documentation/licenses/COPYING.LGPLv2.1 file.

WEV @@ WEV[B "1

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,  
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies  
of this license document, but changing it is not allowed.

#### Preamble

The licenses for most software are designed to take away your  
freedom to share and change it. By contrast, the GNU General Public  
License is intended to guarantee your freedom to share and change free  
software--to make sure the software is free for all its users. This  
General Public License applies to most of the Free Software  
Foundation's software and to any other program whose authors commit to  
using it. (Some other Free Software Foundation software is covered by  
the GNU Lesser General Public License instead.) You can apply it to  
your programs, too.

When we speak of free software, we are referring to freedom, not  
price. Our General Public Licenses are designed to make sure that you  
have the freedom to distribute copies of free software (and charge for  
this service if you wish), that you receive source code or can get it  
if you want it, that you can change the software or use pieces of it  
in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid  
anyone to deny you these rights or to ask you to surrender the rights.

These restrictions translate to certain responsibilities for you if you

distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

## GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or

collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is

void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing

to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

#### NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING

WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

## END OF TERMS AND CONDITIONS

### How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>  
Copyright (C) <year> <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author  
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.  
This is free software, and you are welcome to redistribute it  
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program  
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989  
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, and the entire permission notice in its entirety, including the disclaimer of warranties.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ALL OF WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF NOT ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

x ?"U@,5 @mISmIN<GimCN7g1u|E  
43mI,5WEV @@ WEV @mImImIAmImImI0mImImI\*mImI

A0mImImI...

lost+found...

;9GimCN7g

!"#\$%&'()\*+,-

./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^\_`abcdefghijklmnopqrstuvwxyz{|}~

!"#\$%&'()\*+,-

./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^\_`abcdefghijklmnopqrstuvwxyz{|}~

!"#\$%&'()\*+,-

./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^\_`abcdefghijklmnopqrstuvwxyz{|}~

!"#\$%&'()\*+,-

./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^\_`abcdefghijklmnopqrstuvwxyz{|}~WEV @@

WEV[B "1

This library is free software; you can redistribute it and/or modify it under the terms of the Modified BSD License.

The complete text of the license is available in the `../Documentation/licenses/COPYING.BSD-3` file.

## 1.29 jansson 2.10

### 1.29.1 Available under license :

Copyright (c) 2009-2018 Petri Lehtinen <petri@digip.org>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 1.30 minizip 1.01

## 1.30.1 Available under license :

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 1.31 bouncy-castle 1.65

## 1.31.1 Available under license :

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 1.32 okhttp 4.10.0

## 1.32.1 Available under license :

Note that publicsuffices.gz is compiled from The Public Suffix List:  
[https://publicsuffix.org/list/public\\_suffix\\_list.dat](https://publicsuffix.org/list/public_suffix_list.dat)

It is subject to the terms of the Mozilla Public License, v. 2.0:  
<https://mozilla.org/MPL/2.0/>

# 1.33 protobuf-java 3.20.1

## 1.33.1 Available under license :

No license file was found, but licenses were detected in source scan.

```
// Copyright 2008 Google Inc. All rights reserved.  
// Redistribution and use in source and binary forms, with or without  
// modification, are permitted provided that the following conditions are  
// * Redistributions of source code must retain the above copyright  
// notice, this list of conditions and the following disclaimer.  
// * Redistributions in binary form must reproduce the above  
// copyright notice, this list of conditions and the following disclaimer  
// in the documentation and/or other materials provided with the  
// * Neither the name of Google Inc. nor the names of its  
// this software without specific prior written permission.
```

Found in path(s):

```
* /opt/cola/permits/1483182498_1669147240.6581564/0/protobuf-java-3-20-1-3-jar/google/protobuf/empty.proto  
* /opt/cola/permits/1483182498_1669147240.6581564/0/protobuf-java-3-20-1-3-  
jar/google/protobuf/source_context.proto  
* /opt/cola/permits/1483182498_1669147240.6581564/0/protobuf-java-3-20-1-3-  
jar/google/protobuf/wrappers.proto  
* /opt/cola/permits/1483182498_1669147240.6581564/0/protobuf-java-3-20-1-3-jar/google/protobuf/any.proto  
* /opt/cola/permits/1483182498_1669147240.6581564/0/protobuf-java-3-20-1-3-  
jar/google/protobuf/descriptor.proto  
* /opt/cola/permits/1483182498_1669147240.6581564/0/protobuf-java-3-20-1-3-jar/google/protobuf/struct.proto  
* /opt/cola/permits/1483182498_1669147240.6581564/0/protobuf-java-3-20-1-3-  
jar/google/protobuf/timestamp.proto  
* /opt/cola/permits/1483182498_1669147240.6581564/0/protobuf-java-3-20-1-3-jar/google/protobuf/api.proto  
* /opt/cola/permits/1483182498_1669147240.6581564/0/protobuf-java-3-20-1-3-  
jar/google/protobuf/compiler/plugin.proto  
* /opt/cola/permits/1483182498_1669147240.6581564/0/protobuf-java-3-20-1-3-jar/google/protobuf/type.proto  
* /opt/cola/permits/1483182498_1669147240.6581564/0/protobuf-java-3-20-1-3-jar/google/protobuf/duration.proto  
* /opt/cola/permits/1483182498_1669147240.6581564/0/protobuf-java-3-20-1-3-  
jar/google/protobuf/field_mask.proto
```

No license file was found, but licenses were detected in source scan.

Manifest-Version: 1.0  
Automatic-Module-Name: com.google.protobuf  
Bnd-LastModified: 1650575142706  
Build-Jdk: 1.8.0\_181-google-v7  
Built-By: haberman  
Bundle-Description: Core Protocol Buffers library. Protocol Buffers are a way of encoding structured data in an efficient yet extensible format.  
Bundle-DocURL: <https://developers.google.com/protocol-buffers/>  
Bundle-License: <https://opensource.org/licenses/BSD-3-Clause>  
Bundle-ManifestVersion: 2  
Bundle-Name: Protocol Buffers [Core]  
Bundle-SymbolicName: com.google.protobuf  
Bundle-Version: 3.20.1  
Created-By: Apache Maven Bundle Plugin  
Export-Package: com.google.protobuf;version="3.20.1"  
Import-Package: sun.misc;resolution:=optional,com.google.protobuf;version="[3.20,4)"  
Require-Capability: osgi.ee;filter="(&(osgi.ee=JavaSE)(version=1.7))"  
Tool: Bnd-3.0.0.201509101326

Found in path(s):

\* /opt/cola/permits/1483182498\_1669147240.6581564/0/protobuf-java-3-20-1-3-jar/META-INF/MANIFEST.MF

## 1.34 json-cpp 0.7.0

### 1.34.1 Available under license :

The JsonCpp library's source code, including accompanying documentation, tests and demonstration applications, are licensed under the following conditions...

The author (Baptiste Lepilleur) explicitly disclaims copyright in all jurisdictions which recognize such a disclaimer. In such jurisdictions, this software is released into the Public Domain.

In jurisdictions which do not recognize Public Domain property (e.g. Germany as of 2010), this software is Copyright (c) 2007-2010 by Baptiste Lepilleur, and is released under the terms of the MIT License (see below).

In jurisdictions which recognize Public Domain property, the user of this software may choose to accept it either as 1) Public Domain, 2) under the conditions of the MIT License (see below), or 3) under the terms of dual Public Domain/MIT License conditions described here, as they choose.

The MIT License is about as close to Public Domain as a license can get, and is described in clear, concise terms at:

[http://en.wikipedia.org/wiki/MIT\\_License](http://en.wikipedia.org/wiki/MIT_License)

The full text of the MIT License follows:

=====  
Copyright (c) 2007-2010 Baptiste Lepilleur

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

=====  
(END LICENSE TEXT)

The MIT license is compatible with both the GPL and commercial software, affording one all of the rights of Public Domain with the minor nuisance of being required to keep the above copyright notice and license text in the source code. Note also that by accepting the Public Domain "license" you can re-license your copy using whatever license you like.

## 1.35 libsrtp 2.0.0

### 1.35.1 Available under license :

```
/*
 *
 * Copyright (c) 2001-2006 Cisco Systems, Inc.
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
```

\*  
\* Redistributions of source code must retain the above copyright  
\* notice, this list of conditions and the following disclaimer.  
\*  
\* Redistributions in binary form must reproduce the above  
\* copyright notice, this list of conditions and the following  
\* disclaimer in the documentation and/or other materials provided  
\* with the distribution.  
\*  
\* Neither the name of the Cisco Systems, Inc. nor the names of its  
\* contributors may be used to endorse or promote products derived  
\* from this software without specific prior written permission.  
\*  
\* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS  
\* "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT  
\* LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS  
\* FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE  
\* COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT,  
\* INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES  
\* (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR  
\* SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)  
\* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,  
\* STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)  
\* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED  
\* OF THE POSSIBILITY OF SUCH DAMAGE.  
\*  
\*/

## 1.36 gson 2.8.7

### 1.36.1 Available under license :

No license file was found, but licenses were detected in source scan.

<!--

Copyright 2014-2018 Immutables Authors and Contributors

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

-->

Found in path(s):

\* /opt/cola/permits/1330760950\_1653010360.2168462/0/gson-2-8-7-sources-1-jar/META-INF/maven/org.immutables/gson/pom.xml

No license file was found, but licenses were detected in source scan.

/\*

Copyright 2015 Immutables Authors and Contributors

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

\*/

Found in path(s):

\* /opt/cola/permits/1330760950\_1653010360.2168462/0/gson-2-8-7-sources-1-jar/org/immutables/gson/Gson.java

\* /opt/cola/permits/1330760950\_1653010360.2168462/0/gson-2-8-7-sources-1-

jar/org/immutables/gson/stream/JsonParserReader.java

\* /opt/cola/permits/1330760950\_1653010360.2168462/0/gson-2-8-7-sources-1-

jar/org/immutables/gson/stream/GsonMessageBodyProvider.java

\* /opt/cola/permits/1330760950\_1653010360.2168462/0/gson-2-8-7-sources-1-

jar/org/immutables/gson/stream/JsonGeneratorWriter.java

\* /opt/cola/permits/1330760950\_1653010360.2168462/0/gson-2-8-7-sources-1-

jar/org/immutables/gson/adapters/ExpectedSubtypesAdapter.java

## 1.37 firebase-messaging 31.4.0

### 1.37.1 Available under license :

---

nav\_order: 1

permalink: /

---

# Contributor documentation

This site is a collection of docs and best practices for contributors to Firebase Android SDKs.

It describes how Firebase works on Android and provides guidance on how to build/maintain a Firebase SDK.

## ## New to Firebase?

- [Development Environment Setup]({{ site.baseurl }}{% link onboarding/env\_setup.md %})
- [Creating a new SDK]({{ site.baseurl }}{% link onboarding/new\_sdk.md %})
- [How Firebase works]({{ site.baseurl }}{% link how\_firebase\_works.md %})

---

parent: Best Practices

---

## # Dependency Injection

While [Firebase Components]({{ site.baseurl }}{% link components/components.md %}) provides basic Dependency Injection capabilities for interop between Firebase SDKs, it's not ideal as a general purpose DI framework for a few reasons, to name some:

- \* It's verbose, i.e. requires manually specifying dependencies and constructing instances of components in Component definitions.
- \* It has a runtime cost, i.e. initialization time is linear in the number of Components present in the graph

As a result using [Firebase Components]({{ site.baseurl }}{% link components/components.md %}) is appropriate only for inter-SDK injection and scoping instances per `FirebaseApp`.

On the other hand, manually instantiating SDKs is often tedious, errorprone, and leads to code smells that make code less testable and couples it to the implementation rather than the interface. For more context see [Dependency Injection](https://en.wikipedia.org/wiki/Dependency\_injection) and [Motivation](https://github.com/google/guice/wiki/Motivation).

{: .important }

It's recommended to use [Dagger](https://dagger.dev) for internal dependency injection within the SDKs and [Components]({{ site.baseurl }}{% link components/components.md %}) to inject inter-sdk dependencies that are available only at runtime into the [Dagger Graph](https://dagger.dev/dev-guide/#building-the-graph) via [builder setters](https://dagger.dev/dev-guide/#binding-instances) or [factory arguments](https://dagger.dev/api/latest/dagger/Component.Factory.html).

See: [Dagger docs](https://dagger.dev)

See: [Dagger tutorial](https://dagger.dev/tutorial/)

{: .highlight }

While Hilt is the recommended way to use dagger in Android applications, it's not suitable for SDK/library development.

## ## How to get started

Since [Dagger](https://dagger.dev) does not strictly follow semver and requires the dagger-compiler version to match the dagger library version,

it's not safe to depend on it via a pom level dependency, see [This comment](https://github.com/firebase/firebase-android-sdk/issues/1677#issuecomment-645669608) for context. For this reason in Firebase SDKs we "vendor/repackage" Dagger into the SDK itself under `com.google.firebase.{sdkname}.dagger`. While it incurs in a size increase, it's usually on the order of a couple of KB and is considered negligible.

To use Dagger in your SDK use the following in your Gradle build file:

```
```groovy
plugins {
    id("firebase-vendor")
}

dependencies {
    implementation(libs.javax.inject)
    vendor(libs.dagger.dagger) {
        exclude group: "javax.inject", module: "javax.inject"
    }
    annotationProcessor(libs.dagger.compiler)
}
```
```

## General Dagger setup

As mentioned in [Firebase Components]({{ site.baseurl }}{% link components/components.md %}), all components are scoped per `FirebaseApp` meaning there is a single instance of the component within a given `FirebaseApp`.

This makes it a natural fit to get all inter-sdk dependencies and instantiate the Dagger component inside the `ComponentRegistrar`.

```
```kotlin
class MyRegistrar : ComponentRegistrar {
    override fun getComponents() = listOf(
        Component.builder(MySdk::class.java)
            .add(Dependency.required(FirebaseOptions::class.java))
            .add(Dependency.optionalProvider(SomeInteropDep::class.java))
            .factory(c -> DaggerMySdkComponent.builder()
                .setFirebaseApp(c.get(FirebaseApp::class.java))
                .setSomeInterop(c.getProvider(SomeInteropDep::class.java))
                .build()
            ).getMySdk()
        ).build()
}
```
```

Here's a simple way to define the dagger component:

```

```kotlin
@Component(modules = MySdkComponent.MainModule::class)
@Singleton
interface MySdkComponent {
    // Informs dagger that this is one of the types we want to be able to create
    // In this example we only care about MySdk
    fun getMySdk() : MySdk

    // Tells Dagger that some types are not available statically and in order to create the component
    // it needs FirebaseApp and Provider<SomeInteropDep>
    @Component.Builder
    interface Builder {
        @BindsInstance fun setFirebaseApp(app: FirebaseApp)
        @BindsInstance fun setSomeInterop(interop: com.google.firebase.inject.Provider<SomeInteropDep>)
        fun build() : MySdkComponent
    }

    @Module
    interface MainModule {
        // define module @Provides and @Binds here
    }
}
```

```

The only thing left to do is to properly annotate `MySdk`:

```

```kotlin
@Singleton
class MySdk @Inject constructor(app: FirebaseApp, interopAdapter: MySdkAdapter) {
    fun someMethod() {
        interopAdapter.doInterop()
    }
}

@Singleton
class MySdkInteropAdapter @Inject constructor(private val interop:
com.google.firebase.inject.Provider<SomeInteropDep>) {
    fun doInterop() {
        interop.get().doStuff()
    }
}
```

## Scope

```

Unlike Component, Dagger does not use singleton scope by default and instead injects a new instance of a type at each injection point,

in the example above we want `MySdk` and `MySdkInteropAdapter` to be singletons so they are annotated with `@Singleton`.

See [Scoped bindings](https://dagger.dev/dev-guide/#singletons-and-scoped-bindings) for more details.

### Support multiple instances of the SDK per `FirebaseApp` (multi-resource)

As mentioned in [Firebase Components]({{ site.baseurl }}{% link components/components.md %}), some SDKs support multi-resource mode,

which effectively means that there are 2 scopes at play:

1. `@Singleton` scope that the main `MultiResourceComponent` has.
2. Each instance of the sdk will have its own scope.

```
```mermaid
```

```
flowchart LR
```

```
subgraph FirebaseApp
```

```
direction TB
```

```
subgraph FirebaseComponents
```

```
direction BT
```

```
subgraph GlobalComponents[Outside of SDK]
```

```
direction LR
```

```
    FirebaseOptions
```

```
    SomeInterop
```

```
    Executor["@Background Executor"]
```

```
end
```

```
subgraph DatabaseComponent["@Singleton DatabaseMultiDb"]
```

```
direction TB
```

```
subgraph Singleton["@Singleton"]
```

```
    SomeImpl --> SomeInterop
```

```
    SomeImpl --> Executor
```

```
end
```

```
subgraph Default["@DbScope SDK(default)"]
```

```
    MainClassDefault[FirebaseDatabase] --> SomeImpl
```

```
    SomeOtherImplDefault[SomeOtherImpl] --> FirebaseOptions
```

```
    MainClassDefault --> SomeOtherImplDefault
```

```
end
```

```
subgraph MyDbName["@DbScope SDK(myDbName)"]
```

```
    MainClassMyDbName[FirebaseDatabase] --> SomeImpl
```

```
    SomeOtherImplMyDbName[SomeOtherImpl] --> FirebaseOptions
```

```
    MainClassMyDbName --> SomeOtherImplMyDbName
```

```
end
```

```
end
```

```
end
```

```
end
```

```

classDef green fill:#4db6ac
classDef blue fill:#1a73e8
class GlobalComponents green
class DatabaseComponent green
class Default blue
class MyDbName blue
```

```

As you can see above, `DatabaseMultiDb` and `SomeImpl` are singletons, while `FirebaseDatabase` and `SomeOtherImpl` are scoped per `database name`.

It can be easily achieved with the help of [Dagger's subcomponents](https://dagger.dev/dev-guide/subcomponents).

For example:

```

```kotlin
@Scope
annotation class DbScope

@Component(modules = DatabaseComponent.MainModule::class)
interface DatabaseComponent {
    fun getMultiDb() : DatabaseMultiDb

    @Component.Builder
    interface Builder {
        // usual setters for Firebase component dependencies
        // ...
        fun build() : DatabaseComponent
    }

    @Module(subcomponents = DbInstanceComponent::class)
    interface MainModule { }

    @Subcomponent(modules = DbInstanceComponent.InstanceModule::class)
    @DbScope
    interface DbInstanceComponent {
        fun factory() : Factory

        @Subcomponent.Factory
        interface Factory {
            fun create(@BindsInstance @Named("dbName") dbName: String)
        }
    }

    @Module
    interface InstanceModule {
        // ...
    }
}
```

```

```
}  
}  
...
```

Annotating `FirebaseDatabase`:

```
```kotlin  
@DbScope  
class FirebaseDatabase @Inject constructor(options: FirebaseOptions, @Named dbName: String) {  
    // ...  
}  
...
```

Implementing `DatabaseMultiDb`:

```
```kotlin  
@Singleton  
class DatabaseMultiDb @Inject constructor(private val factory: DbInstanceComponent.Factory) {  
    private val instances = mutableMapOf<String, FirebaseDatabase>()
```

```
    @Synchronized  
    fun get(dbName: String) : FirebaseDatabase {  
        if (!instances.containsKey(dbName)) {  
            mInstances.put(dbName, factory.create(dbName))  
        }  
        return mInstances.get(dbName);  
    }  
}  
...
```

```
---  
parent: Onboarding  
---
```

```
# Creating a new Firebase SDK  
{: .no_toc}
```

```
1. TOC  
{:toc}
```

Want to create a new SDK in  
[firebase/firebase-android-sdk](https://github.com/firebase/firebase-android-sdk)?  
Read on.

```
{:toc}
```

```
## Repository layout and Gradle
```

```
[firebase/firebase-android-sdk](https://github.com/firebase/firebase-android-sdk)
```

uses a multi-project Gradle build to organize the different libraries it hosts. As a consequence, each project/product within this repo is hosted under its own subdirectory with its respective build file(s).

```
```bash
firebase-android-sdk
buildSrc
appcheck
  firebase-appcheck
  firebase-appcheck-playintegrity
firebase-annotations
firebase-common
  firebase-common.gradle # note the name of the build file
ktx
  ktx.gradle.kts # it can also be kts
build.gradle # root project build file.
```
```

Most commonly, SDKs are located as immediate child directories of the root directory, with the directory name being the exact name of the Maven artifact ID the library will have once released. e.g. `firebase-common` directory hosts code for the `com.google.firebase:firebase-common` SDK.

```
{: .warning }
```

Note that the build file name for any given SDK is not `build.gradle` or `build.gradle.kts` but rather mirrors the name of the sdk, e.g.

```
`firebase-common/firebase-common.gradle` or `firebase-common/firebase-common.gradle.kts`.
```

All of the core Gradle build logic lives in `buildSrc` and is used by all SDKs.

SDKs can be grouped together for convenience by placing them in a directory of choice.

### ## Creating an SDK

Let's say you want to create an SDK named `firebase-foo`

1. Create a directory called `firebase-foo`.
1. Create a file `firebase-foo/firebase-foo.gradle.kts`.
1. Add `firebase-foo` line to `subprojects.cfg` at the root of the tree.

### Update `firebase-foo.gradle.kts` with the following content

```
<details open markdown="block">
```

```
<summary>
```

```
  firebase-foo.gradle.kts
```

```
</summary>
```

```

```kotlin
plugins {
    id("firebase-library")
    // Uncomment for Kotlin
    // id("kotlin-android")
}

firebaseLibrary {
    // enable this only if you have tests in `androidTest`.
    testLab.enabled = true
    publishSources = true
    publishJavadoc = true
}

android {
    val targetSdkVersion : Int by rootProject
    val minSdkVersion : Int by rootProject

    compileSdk = targetSdkVersion
    defaultConfig {
        namespace = "com.google.firebase.foo"
        // change this if you have custom needs.
        minSdk = minSdkVersion
        targetSdk = targetSdkVersion
        testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
    }

    testOptions.unitTests.isIncludeAndroidResources = true
}

dependencies {
    implementation(project(":firebase-common"))
    implementation(project(":firebase-components"))
}
```

```

...

</details>

### Create `src/main/AndroidManifest.xml` with the following content:

```

<details open markdown="block">
<summary>
    src/main/AndroidManifest.xml
</summary>
```xml
<?xml version="1.0" encoding="utf-8"?>
<!-- Copyright {{ 'now' | date: "%Y" }} Google LLC -->
<!-- -->

```

```

<!-- Licensed under the Apache License, Version 2.0 (the "License"); -->
<!-- you may not use this file except in compliance with the License. -->
<!-- You may obtain a copy of the License at -->
<!-- -->
<!-- http://www.apache.org/licenses/LICENSE-2.0 -->
<!-- -->
<!-- Unless required by applicable law or agreed to in writing, software -->
<!-- distributed under the License is distributed on an "AS IS" BASIS, -->
<!-- WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. -->
<!-- See the License for the specific language governing permissions and -->
<!-- limitations under the License. -->

<manifest xmlns:android="http://schemas.android.com/apk/res/android">
  <application>
    <service android:name="com.google.firebase.components.ComponentDiscoveryService"
      android:exported="false">
      <meta-data
        android:name="com.google.firebase.components:com.google.firebase.foo.FirebaseFooRegistrar"
        android:value="com.google.firebase.components.ComponentRegistrar" />
    </service>
  </application>
</manifest>

```

```

</details>

```

```

### Create `com.google.firebase.foo.FirebaseFoo`

```

```

For Kotlin

```

```

<details open markdown="block">
  <summary>
    src/main/kotlin/com/google/firebase/foo/FirebaseFoo.kt
  </summary>

```

```

```kotlin
class FirebaseFoo {
  companion object {
    @JvmStatic
    val instance: FirebaseFoo
    get() = getInstance(Firebase.app)

    @JvmStatic fun getInstance(app: FirebaseApp): FirebaseFoo = app.get(FirebaseFoo::class.java)
  }
}
```

```

```

</details>

```

For Java

<details markdown="block">

<summary>

src/main/java/com/google/firebase/foo/FirebaseFoo.java

</summary>

```
```java
public class FirebaseFoo {
    public static FirebaseFoo getInstance() {
        return getInstance(FirebaseApp.getInstance());
    }
    public static FirebaseFoo getInstance(FirebaseApp app) {
        return app.get(FirebaseFoo.class);
    }
}
```
```

</details>

### Create `com.google.firebase.foo.FirebaseFooRegistrar`

For Kotlin

<details open markdown="block">

<summary>

src/main/kotlin/com/google/firebase/foo/FirebaseFooRegistrar.kt

</summary>

{: .warning }

You should strongly consider using [Dependency Injection]({{ site.baseurl }}{% link best\_practices/dependency\_injection.md %})

to instantiate your sdk instead of manually constructing its instance in the `factory()` below.

```
```kotlin
class FirebaseFooRegistrar : ComponentRegistrar {
    override fun getComponents() =
        listOf(
            Component.builder(FirebaseFoo::class.java).factory { container -> FirebaseFoo() }.build(),
            LibraryVersionComponent.create("fire-foo", BuildConfig.VERSION_NAME)
        )
}
```
```

</details>

For Java

<details markdown="block">

<summary>

src/main/java/com/google/firebase/foo/FirebaseFooRegistrar.java

</summary>

```
```java
public class FirebaseFooRegistrar implements ComponentRegistrar {
    @Override
    public List<Component<?>> getComponents() {
        return Arrays.asList(
            Component.builder(FirebaseFoo.class).factory(c -> new FirebaseFoo()).build(),
            LibraryVersionComponent.create("fire-foo", BuildConfig.VERSION_NAME));
    }
}
```
```

</details>

Continue to [How Firebase works]({{ site.baseurl }}{% link how\_firebase\_works.md %}).

---

has\_children: true  
permalink: /best\_practices/  
nav\_order: 5

---

# Best Practices

title: Contributor documentation

description: Documentation and best practices for Android SDK development

logo: "https://firebase.google.com/downloads/brand-guidelines/SVG/logo-logomark.svg"

remote\_theme: just-the-docs/just-the-docs@v0.4.0.rc3

plugins:

- jekyll-remote-theme

color\_scheme: light

# Aux links for the upper right navigation

aux\_links:

"SDK Github Repo":

- "///github.com/firebase/firebase-android-sdk"

# Enable or disable the site search

# Supports true (default) or false

search\_enabled: true

search:

# Split pages into sections that can be searched individually

# Supports 1 - 6, default: 2

heading\_level: 6

# Maximum amount of previews per search result

# Default: 3

```
previews: 2
# Maximum amount of words to display before a matched word in the preview
# Default: 5
preview_words_before: 3
# Maximum amount of words to display after a matched word in the preview
# Default: 10
preview_words_after: 3
# Set the search token separator
# Default: /\s\-/+/
# Example: enable support for hyphenated search words
tokenizer_separator: /\s/+/
# Display the relative url in search results
# Supports true (default) or false
rel_url: true
# Enable or disable the search button that appears in the bottom right corner of every page
# Supports true or false (default)
button: false

mermaid:
# Version of mermaid library
# Pick an available version from https://cdn.jsdelivr.net/npm/mermaid/
version: "9.2.2"
# Put any additional configuration, such as setting the theme, in _includes/mermaid_config.js

# Enable or disable heading anchors
heading_anchors: true
permalink: pretty

callouts_level: quiet
callouts:
highlight:
  color: yellow
important:
  title: Important
  color: blue
new:
  title: New
  color: green
note:
  title: Note
  color: purple
warning:
  title: Warning
  color: red

# Back to top link
back_to_top: true
back_to_top_text: "Back to top"
```

```

# Footer last edited timestamp
last_edit_timestamp: true # show or hide edit time - page must have `last_modified_date` defined in the frontmatter
last_edit_time_format: "%b %e %Y at %I:%M %p" # uses ruby's time format: https://ruby-doc.org/stdlib-2.7.0/libdoc/time/rdoc/Time.html

# Footer "Edit this page on GitHub" link text
gh_edit_link: true # show or hide edit this page link
gh_edit_link_text: "Edit this page on GitHub"
gh_edit_repository: "https://github.com/firebase/firebase-android-sdk" # the github URL for your repo
gh_edit_branch: "master" # the branch that your docs is served from
gh_edit_source: "contributor-docs" # the source that your files originate from
gh_edit_view_mode: "edit" # "tree" or "edit" if you want the user to jump into the editor immediately
name: Copyright check

on: pull_request

concurrency:
  group: ${{ github.workflow }}-${{ github.event.pull_request.number || github.ref }}
  cancel-in-progress: true

jobs:
  copyright-check:
    runs-on: ubuntu-22.04
    steps:
      - uses: actions/checkout@v3.0.2
      - uses: actions/setup-python@v2
        with:
          python-version: '3.9'
      - run: |
          pip install -e "ci/fireci"
      - run: |
          fireci copyright_check \
            -e py          \
            -e gradle      \
            -e java        \
            -e kt          \
            -e groovy      \
            -e sh          \
            -e proto
Test license
---
parent: Firebase Components
---

# Dynamic Module Support

```

TODO

---

nav\_order: 3

---

# How Firebase Works

## Background

### Eager Initialization

One of the biggest strengths for Firebase clients is the ease of integration. In a common case, a developer has very few things to do to integrate with Firebase. There is no need to initialize/configure Firebase at runtime. Firebase automatically initializes at application start and begins providing value to developers. A few notable examples:

- \* `Analytics` automatically tracks app events
- \* `Firebase Performance` automatically tracks app startup time, all network requests and screen performance
- \* `Crashlytics` automatically captures all application crashes, ANRs and non-fatals

This feature makes onboarding and adoption very simple. However, comes with the great responsibility of keeping the application snappy. We shouldn't slow down application startup for 3p developers as it can stand in the way of user adoption of their application.

### Automatic Inter-Product Discovery

When present together in an application, Firebase products can detect each other and automatically provide additional functionality to the developer, e.g.:

- \* `Firestore` automatically detects `Auth` and `AppCheck` to protect read/write access to the database
- \* `Crashlytics` integrates with `Analytics`, when available, to provide additional insights into the application behavior and enables safe app rollouts

## FirebaseApp at the Core of Firebase

Regardless of what Firebase SDKs are present in the app, the main initialization point of Firebase is `FirebaseApp`. It acts as a container for all SDKs, manages their configuration, initialization and lifecycle.

### Initialization

`FirebaseApp` gets initialized with the help of `FirebaseApp#initializeApp()`. This happens [automatically at app startup](<https://firebase.blog/posts/2016/12/how-does-firebase-initialize-on-android>) or manually by the developer.

During initialization, `FirebaseApp` discovers all Firebase SDKs present in the app, determines the dependency graph between products(for inter-product functionality) and initializes `eager` products that need to start immediately, e.g. `Crashlytics` and `FirebasePerformance`.

### Firebase Configuration

`FirebaseApp` contains Firebase configuration for all products to use, namely `FirebaseOptions`, which tells Firebase which `Firebase` project to talk to, which real-time database to use, etc.

### ### Additional Services/Components

In addition to `FirebaseOptions`, `FirebaseApp` registers additional components that product SDKs can request via dependency injection. To name a few:

- \* `android.content.Context` (Application context)
- \* [Common Executors]({{ site.baseurl }}{% link components/executors.md %})
- \* `FirebaseOptions`
- \* Various internal components

### ## Discovery and Dependency Injection

There are multiple considerations that lead to the current design of how Firebase SDKs initialize.

1. Certain SDKs need to initialize at app startup.
2. SDKs have optional dependencies on other products that get enabled when the developer adds the dependency to their app.

To enable this functionality, Firebase uses a runtime discovery and dependency injection framework [firebase-components](https://github.com/firebase/firebase-android-sdk/tree/master/firebase-components).

To integrate with this framework SDKs register the components they provide via a `ComponentRegistrar` and declare any dependencies they need to initialize, e.g.

```
```java
public class MyRegistrar implements ComponentRegistrar {
    @Override
    public List<Component<?>> getComponents() {
        return Arrays.asList(
            // declare the component
            Component.builder(MyComponent.class)
                // declare dependencies
                .add(Dependency.required(Context.class))
                .add(Dependency.required(FirebaseOptions.class))
                .add(Dependency.optionalProvider(InternalAuthProvider.class))
                // let the runtime know how to create your component.
                .factory(
                    diContainer ->
                    new MyComponent(
                        diContainer.get(Context.class),
                        diContainer.get(FirebaseOptions.class),
                        diContainer.get(InternalAuthProvider.class)))
                .build());
        }
    }
}
```

...

This registrar is then registered in `AndroidManifest.xml` of the SDK and is used by `FirebaseApp` to discover all components and construct the dependency graph.

More details in [Firebase Components]({{ site.baseurl }}{% link components/components.md %}).

---

parent: Onboarding

---

## # Development Environment Setup

This page describes software and configuration required to work on code in the [Firebase/firebase-android-sdk](https://github.com/firebase/firebase-android-sdk) repository.

{:toc}

## ## JDK

The currently required version of the JDK is `11`. Any other versions are unsupported and using them could result in build failures.

## ## Android Studio

In general, the most recent version of Android Studio should work. The version that is tested at the time of this writing is `Dolphin | 2021.3.1`.

Download it here:

[Download Android Studio](https://developer.android.com/studio)

## ## Emulators

If you plan to run tests on emulators (you should), you should be able to install them directly from Android Studio's AVD manager.

## ## Github (Googlers Only)

To onboard and get write access to the github repository you need to have a github account fully linked with [go/github](http://go/github).

File a bug using this

[bug template](http://b/issues/new?component=312729&template=1016566) and wait for access to be granted.

After that configure github keys as usual using this

[Github documentation page](https://docs.github.com/en/github/authenticating-to-github/connecting-to-github-with-ssh).

## ## Importing the repository

1. Clone the repository with `git clone --recurse-submodules git@github.com:firebase/firebase-android-sdk.git`.`
1. Open Android Studio and click "Open an existing project".  
![Open an existing project](as\_open\_project.png)
1. Find the `firebase-android-sdk`` directory and open.
1. To run integration/device tests you will need a `google-services.json`` file.

---

```
has_children: true
permalink: /components/
nav_order: 4
```

---

## # Firebase Components

```
{: .no_toc}
```

### 1. TOC

```
{:toc}
```

Firebase is known for being easy to use and requiring no/minimal configuration at runtime. Just adding SDKs to the app makes them discover each other to provide additional functionality, e.g. `Firestore`` automatically integrates with `Auth`` if present in the app.

- \* Firebase SDKs have required and optional dependencies on other Firebase SDKs
- \* SDKs have different initialization requirements, e.g. `Analytics`` and `Crashlytics`` must be initialized upon application startup, while some are initialized on demand only.

To accommodate these requirements Firebase uses a component model that discovers SDKs present in the app, determines their dependencies and provides them to dependent SDKs via a `Dependency Injection`` mechanism.

This page describes the aforementioned Component Model, how it works and why it's needed.

## ## Design Considerations

### ### Transparent/invisible to 3p Developers

To provide good developer experience, we don't want developers to think about how SDKs work and interoperate internally.

Instead we want our SDKs to have a simple API surface that hides all of the internal details.

Most products have an API surface that allows developers to get an instance of a given SDK via `FirestoreFoo.getInstance()`` and start using it right away.

### ### Simple to use and integrate with for component developers

- \* The component model is lightweight in terms of integration effort. It is not opinionated on how components are

structured.

- \* The component model should require as little cooperation from components runtime as possible.
- \* It provides component developers with an API that is easy to use correctly, and hard to use incorrectly.
- \* Does not sacrifice testability of individual components in isolation

### Performant at startup and initialization

The runtime does as little work as possible during initialization.

## What is a Component?

A Firebase Component is an entity that:

- \* Implements one or more interfaces
- \* Has a list of dependencies(required or optional). See [Dependencies]({{ site.baseurl }}{% link components/dependencies.md %})
- \* Has initialization requirements(e.g. eager in default app)
- \* Defines a factory creates an instance of the components interface given it's dependencies.  
(In other words describes how to create the given component.)

Example:

```
```java
// Defines a component that is registered as both `FirebaseAuth` and `InternalAuthProvider`.
Component<FirebaseAuth> auth = Component.builder(FirebaseAuth.class, InternalAuthProvider.class)
    // Declares dependencies
    .add(Dependency.required(FirebaseOptions.class))
    // Defines a factory
    .factory(container -> new FirebaseAuth(container.get(FirebaseOptions.class)))
    .eagerInDefaultApp() // alwaysEager() or lazy(), lazy is the default.
    .build()
```
```

All components are singletons within a Component Container(e.g. one instance per FirebaseAuth). There are however SDKs that need the ability to expose multiple objects per FirebaseAuth, for example RTBD(as well as Storage and Firestore) has multidb support which allows developers to access one or more databases within one FirebaseAuth. To address this requirement, SDKs have to register their components in the following form(or similar):

```
```java
// This is the singleton holder of different instances of FirebaseDatabase.
interface RtdbComponent {
    FirebaseDatabase getDefault();
    FirebaseDatabase get(String databaseName);
}
```
```

As you can see in the previous section, components are just values and don't have any behavior per se,

essentially they are just blueprints of how to create them and what dependencies they need.

So there needs to be some `ComponentRuntime` that can discover and wire them together into a dependency graph, in order to do that, there needs to be an agreed upon location where SDKs can register the components they provide.

The next 2 sections describe how it's done.

## ## Component Registration

In order to define the `Components` an SDK provides, it needs to define a class that implements `ComponentRegistrar`, this class contains all component definitions the SDK wants to register with the runtime:

```
```java
public class MyRegistrar implements ComponentRegistrar {
    /// Returns a one or more Components that will be registered in
    /// FirebaseAuth and participate in dependency resolution and injection.
    @Override
    public Collection<FirebaseComponent<?>> getComponents() {
        Arrays.asList(Component.builder(MyType.class)
            /* ... */
            .build());
    }
}
```
```

## ## Component Discovery

In addition to creating the `ComponentRegistrar` class, SDKs also need to add them to their `AndroidManifest.xml` under `ComponentDiscoveryService`:

```
```xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android">
  <application>
    <service android:name="com.google.firebase.components.ComponentDiscoveryService"
      android:exported="false">
      <meta-data
        android:name="com.google.firebase.components:com.google.firebase.foo.FirebaseFooRegistrar"
        android:value="com.google.firebase.components.ComponentRegistrar" />
    </service>
  </application>
</manifest>
```
```

When the final app is built, manifest registrar entries will all end up inside the above `service` as metadata key-value pairs.

At this point `FirebaseApp` will instantiate them and use the `ComponentRuntime` to construct the component graph.

## ## Dependency resolution and initialization

### ### Definitions and constraints

\* \*\*Component A depends on Component B\*\* if `B` depends on an `interface` that `A` implements.

\* \*\*For any Interface I, only one component is allowed to implement I\*\* (with the exception of [Set Dependencies]({{ site.baseurl }}{% link components/dependencies.md % }#set-dependencies)). If this invariant is violated, the container will fail to start at runtime.

\* \*\*There must not be any dependency cycles\*\* among components. See Dependency Cycle Resolution on how this limitation can be mitigated

\* \*\*Components are initialized lazily by default\*\* (unless a component is declared eager) and are initialized when requested by an application either directly or transitively.

The initialization phase of the `FirebaseApp` will consist of the following steps:

1. Get a list of available `FirebaseComponents` that were discovered by the Discovery mechanism
2. Topologically sort components based on their declared dependencies - failing if a dependency cycle is detected or multiple implementations are registered for any interface.
3. Store a map of {iface -> `ComponentFactory`} so that components can be instantiated on demand (Note that component instantiation does not yet happen)
4. Initialize EAGER components or schedule them to initialize on device unlock, if in direct boot mode.

### ### Initialization example

Below is an example illustration of the state of the component graph after initialization:

```
```mermaid
flowchart TD
  Analytics --> Installations
  Auth --> Context
  Auth --> FirebaseOptions
  Context[android.os.Context]
  Crashlytics --> Installations
  Crashlytics --> FirebaseApp
  Crashlytics --> FirebaseOptions
  Crashlytics -. -> Analytics
  Crashlytics --> Context
  Database -. -> Auth
  Database --> Context
  Database --> FirebaseApp
  Database --> FirebaseOptions
  Firestore -. -> Auth
  Messaging --> Installations
  Messaging --> FirebaseOptions
```
```

```
Messaging --> Context
RemoteConfig --> FirebaseApp
RemoteConfig --> Context
RemoteConfig --> Installations
```

```
classDef eager fill:#4db66e,stroke:#4db6ac,color:#000;
classDef transitive fill:#4db6ac,stroke:#4db6ac,color:#000;
classDef always fill:#1a73e8,stroke:#7baaf7,color:#fff;
```

```
class Analytics eager
class Crashlytics eager
class Context always
class FirebaseOptions always
class FirebaseApp always
class Installations transitive
...

```

There are **2** explicitly eager components in this example: ``Crashlytics`` and ``Analytics``.

These components are initialized when ``FirebaseApp`` is initialized. ``Installations`` is initialized eagerly because eager components depends on it(see Prefer Lazy dependencies to avoid this as much as possible).

``FirebaseApp``, ``FirebaseOptions`` and ``Android Context`` are always present in the Component Container and are considered initialized as well.

\*The rest of the components are left uninitialized and will remain so until the client application requests them or an eager

component initializes them by using a Lazy dependency.\*

For example, if the application calls ``FirebaseDatabase.getInstance()``, the container will initialize ``Auth`` and ``Database``

and will return ``Database`` to the user.

```
### Support multiple instances of the SDK per `FirebaseApp` (multi-resource)
```

Some SDKs support multi-resource mode of operation, where it's possible to create more than one instance per ``FirebaseApp``.

Examples:

\* RTDB allows more than one database in a single Firebase project, so it's possible to instantiate one instance of the sdk per database

```
```kotlin
val rtdbOne = Firebase.database(app) // uses default database
val rtdbTwo = Firebase.database(app, "dbName")
```
```

\* Firestore, functions, and others support the same usage pattern

To allow for that, such SDKs register a singleton "MultiResource" [Firebase component]({{ site.baseurl }}{% link components/components.md %}), which creates instances per resource(e.g. db name).

Example

```
```kotlin
class DatabaseComponent(private val app: FirebaseApp, private val tokenProvider: InternalTokenProvider) {
    private val instances: MutableMap<String, FirebaseDatabase> = new HashMap<>();

    @Synchronized
    fun get(dbName: String) : FirebaseDatabase {
        if (!instances.containsKey(dbName)) {
            instances.put(dbName, FirebaseDatabase(app, tokenProvider, dbName))
        }
        return instances.get(dbName);
    }
}

class FirebaseDatabase(
    app: FirebaseApp,
    tokenProvider: InternalTokenProvider,
    private val String dbName)

    companion object {
        fun getInstance(app : FirebaseApp) = getInstance("default")
        fun getInstance(app : FirebaseApp, dbName: String) =
            app.get(DatabaseComponent::class.java).get("default")
    }
}
```

---
has_children: true
permalink: /onboarding/
nav_order: 2
---

# Onboarding
---
parent: Firebase Components
---

# Executors
{: .no_toc}

1. TOC
{:toc}
```

## ## Intro

OS threads are a limited resource that needs to be used with care. In order to minimize the number of threads used by Firebase as a whole and to increase resource sharing Firebase Common provides a set of standard [executors](https://developer.android.com/reference/java/util/concurrent/Executor) and [coroutine dispatchers](https://kotlinlang.org/api/kotlinx.coroutines/kotlinx-coroutines-core/kotlinx.coroutines/-coroutine-dispatcher/) for use by all Firebase SDKs.

These executors are available as components and can be requested by product SDKs as component dependencies.

Example:

```
```java
public class MyRegistrar implements ComponentRegistrar {
    public List<Component<?>> getComponents() {
        Qualified<Executor> backgroundExecutor = Qualified.qualified(Background.class, Executor.class);
        Qualified<ExecutorService> liteExecutorService = Qualified.qualified(Lightweight.class, ExecutorService.class);

        return Collections.singletonList(
            Component.builder(MyComponent.class)
                .add(Dependency.required(backgroundExecutor))
                .add(Dependency.required(liteExecutorService))
                .factory(c -> new MyComponent(c.get(backgroundExecutor), c.get(liteExecutorService)))
                .build());
    }
}
```
```

All executors (with the exception of `@UiThread`) are available as the following interfaces:

- \* `Executor`
- \* `ExecutorService`
- \* `ScheduledExecutorService`
- \* `CoroutineDispatcher`

`@UiThread` is provided only as a plain `Executor`.

## ### Validation

All SDKs have a custom linter check that detects creation of thread pools and threads, this is to ensure SDKs use the above executors instead of creating their own.

## ## Choose the right executor

Use the following diagram to pick the right executor for the task you have at hand.

```

```mermaid
flowchart TD
  Start[Start] --> DoesBlock{Does it block?}
  DoesBlock -->|No| NeedUi{Does it need to run\n on UI thread?}
  NeedUi -->|Yes| UiExecutor[[UiThread Executor]]
  NeedUi -->|No| TakesLong{Does it take more than\n 10ms to execute?}
  TakesLong -->|No| LiteExecutor[[Lightweight Executor]]
  TakesLong -->|Yes| BgExecutor[[Background Executor]]
  DoesBlock -->|Yes| DiskIO{Does it block only\n on disk IO?}
  DiskIO -->|Yes| BgExecutor
  DiskIO -->|No| BlockExecutor[[Blocking Executor]]

```

```

classDef start fill:#4db6ac,stroke:#4db6ac,color:#000;
class Start start

```

```

classDef condition fill:#f8f9fa,stroke:#bdc1c6,color:#000;
class DoesBlock condition;
class NeedUi condition;
class TakesLong condition;
class DiskIO condition;

```

```

classDef executor fill:#1a73e8,stroke:#7baaf7,color:#fff;
class UiExecutor executor;
class LiteExecutor executor;
class BgExecutor executor;
class BlockExecutor executor;

```

```

```

```

### ### UiThread

Used to schedule tasks on application's UI thread, internally it uses a Handler to post runnables onto the main looper.

Example:

```

```java
// Java
Qualified<Executor> uiExecutor = qualifiedUiThread.class, Executor.class);
```

```

```

```kotlin
// Kotlin
Qualified<CoroutineDispatcher> dispatcher = qualifiedUiThread::class.java, CoroutineDispatcher::class.java);
```

```

### ### Lightweight

Use for tasks that never block and don't take too long to execute. Backed by a thread pool of N threads where N is the amount of parallelism available on the device (number of CPU cores)

Example:

```
```java
// Java
Qualified<Executor> liteExecutor = qualified(Lightweight.class, Executor.class);
```

```kotlin
// Kotlin
Qualified<CoroutineDispatcher> dispatcher = qualified(Lightweight::class.java, CoroutineDispatcher::class.java);
```
```

### ### Background

Use for tasks that may block on disk IO (use `@Blocking` for network IO or blocking on other threads). Backed by 4 threads.

Example:

```
```java
// Java
Qualified<Executor> bgExecutor = qualified(Background.class, Executor.class);
```

```kotlin
// Kotlin
Qualified<CoroutineDispatcher> dispatcher = qualified(Background::class.java, CoroutineDispatcher::class.java);
```
```

### ### Blocking

Use for tasks that can block for arbitrary amounts of time, this includes network IO.

Example:

```
```java
// Java
Qualified<Executor> blockingExecutor = qualified(Blocking.class, Executor.class);
```

```kotlin
// Kotlin
Qualified<CoroutineDispatcher> dispatcher = qualified(Blocking::class.java, CoroutineDispatcher::class.java);
```
```

### Other executors

#### Direct executor

```
{: .warning }
```

Prefer `@Lightweight` instead of using direct executor as it could cause dead locks and stack overflows.

For any trivial tasks that don't need to run asynchronously

Example:

```
```kotlin
FirebaseExecutors.directExecutor()
```
```

#### Sequential Executor

When you need an executor that runs tasks sequentially and guarantees any memory access is synchronized prefer to use a sequential executor instead of creating a `newSingleThreadedExecutor()`.

Example:

```
```java
// Pick the appropriate underlying executor using the chart above
Qualified<Executor> bgExecutor = qualified(Background.class, Executor.class);
// ...
Executor sequentialExecutor = FirebaseExecutors.newSequentialExecutor(c.get(bgExecutor));
```
```

## Proper Kotlin usage

A `CoroutineContext` should be preferred when possible over an explicit `Executor` or `CoroutineDispatcher`. You should only use an `Executor` at the highest (or inversely the lowest) level of your implementations. Most classes should not be concerned with the existence of an `Executor`.

Keep in mind that you can combine `CoroutineContext` with other `CoroutineScope` or `CoroutineContext`. And that all `suspend` functions inherit their `coroutineContext`:

```
```kotlin
suspend fun createSession(): Session {
    val context = backgroundDispatcher.coroutineContext + coroutineContext
    return Session(context)
}
```
```

To learn more, you should give the following Kotlin wiki page a read:

[Coroutine context and dispatchers](https://kotlinlang.org/docs/coroutine-context-and-dispatchers.html#dispatchers-and-threads)

## Testing

### Using Executors in tests

`@Lightweight` and `@Background` executors have `StrictMode` enabled and throw exceptions on violations. For example trying to do Network IO on either of them will throw.

With that in mind, when it comes to writing tests, prefer to use the common executors as opposed to creating your own thread pools. This will ensure that your code uses the appropriate executor and does not slow down all of Firebase by using the wrong one.

To do that, you should prefer relying on Components to inject the right executor even in tests. This will ensure your tests are always using the executor that is actually used in your SDK build. If your SDK uses Dagger, see [Dependency Injection]({{ site.baseurl }}{% link best\_practices/dependency\_injection.md %}) and [Dagger's testing guide](https://dagger.dev/dev-guide/testing).

When the above is not an option, you can use `TestOnlyExecutors`, but make sure you're testing your code with the same executor that is used in production code:

```
```kotlin
dependencies {
    // ...
    testImplementation(project(":integ-testing"))
    // or
    androidTestImplementation(project(":integ-testing"))
}
```
```

This gives access to

```
```java
TestOnlyExecutors.ui();
TestOnlyExecutors.background();
TestOnlyExecutors.blocking();
TestOnlyExecutors.lite();
```
```

### Policy violations in tests

Unit tests require [Robolectric](https://github.com/robolectric/robolectric) to function correctly, and this comes with a major drawback; no policy validation.

Robolectric supports `StrictMode` - but does not provide the backing for its policy mechanisms to fire on violations. As such, you'll be able to do things like using `TestOnlyExecutors.background()` to execute blocking actions; usage

that would have otherwise crashed in a real application.

Unfortunately, there is no easy way to fix this for unit tests. You can get around the issue by moving the tests to an emulator (integration tests)- but those can be more expensive than your standard unit test, so you may want to take that into consideration when planning your testing strategy.

### ### StandardTestDispatcher support

The `[kotlin.coroutines.test](https://kotlin.github.io/kotlinx.coroutines/kotlinx-coroutines-test/)` library provides support for a number of different mechanisms in tests. Some of the more famous features include:

- `[advanceUntilIdle](https://kotlin.github.io/kotlinx.coroutines/kotlinx-coroutines-test/kotlinx.coroutines.test/-test-coroutine-scheduler/advance-until-idle.html)`
- `[advanceTimeBy](https://kotlin.github.io/kotlinx.coroutines/kotlinx-coroutines-test/kotlinx.coroutines.test/-test-coroutine-scheduler/advance-time-by.html)`
- `[runCurrent](https://kotlin.github.io/kotlinx.coroutines/kotlinx-coroutines-test/kotlinx.coroutines.test/-test-coroutine-scheduler/run-current.html)`

These features are all backed by ``StandardTestDispatcher``, or more appropriately, the ``TestScope`` provided in a ``runTest`` block.

Unfortunately, ``TestOnlyExecutors`` does not natively bind with ``TestScope``. Meaning, should you use ``TestOnlyExecutors`` in your tests- you won't be able to utilize the features provided by ``TestScope``:

```
```kotlin
@Test
fun doesStuff() = runTest {
    val scope = CoroutineScope(TestOnlyExecutors.background().asCoroutineDispatcher())
    scope.launch {
        // ... does stuff
    }

    runCurrent() // doesn't invoke scope ??
}
```
```

To help fix this, we provide an extension method on ``TestScope`` called ``firebaseExecutors``. It facilitates the binding of ``TestOnlyExecutors`` with the current ``TestScope``.

For example, here's how you could use this extension method in a test:

```
```kotlin
@Test
fun doesStuff() = runTest {
```

```

    val scope = CoroutineScope(firebaseExecutors.background)
    scope.launch {
        // ... does stuff
    }

    runCurrent()
}
...
---
parent: Firebase Components
---

# Dependencies
{: .no_toc}

1. TOC
{:toc}

```

This page gives an overview of the different dependency types supported by the Components Framework.

## ## Background

As discussed in [\[Firebase Components\]](#)([{{ site.baseurl }}{% link components/components.md %}](#)), in order for a `Component` to be injected with the things it needs to function, it has to declare its dependencies. These dependencies are then made available and injected into `Components` at runtime.

Firebase Components provide different types of dependencies.

## ## Lazy vs Eager dependencies

When it comes to initialize a component, there are 2 ways of provide its dependencies.

### ### Direct Injection

With this type of injection, the component gets an instance of its dependency directly.

```

```kotlin
class MyComponent(private val dep : MyDep) {
    fun someMethod() {
        dep.use();
    }
}
```

```

As you can see above the component's dependency is passed by value directly, which means that the dependency needs to be fully initialized before it's handed off to the requesting component. As a result `MyComponent` may have to pay the cost of initializing `MyDep` just to be created.

### ### Lazy/Provider Injection

With this type of injection, instead of getting an instance of the dependency directly, the dependency is passed into the `Component` with the help of a `com.google.firebase.inject.Provider`

```
```java
public interface Provider<T> { T get(); }
```

```kotlin
class MyComponent(private val dep : Provider<MyDep>) {
    fun someMethod() {
        // Since all components are singletons, each call to
        // get() will return the same instance.
        dep.get().use();
    }
}
```
```

On the surface this does not look like a big change, but it has an important side effect. In order to create an instance of `MyComponent`, we don't need to initialize `MyDep` anymore. Instead, initialization can be delayed until `MyDep` is actually used.

It is also beneficial to use a `Provider` in the context of [Play's dynamic feature delivery](https://developer.android.com/guide/playcore/feature-delivery).

See [Dynamic Module Support]({{ site.baseurl }}{% link components/dynamic\_modules.md %}) for more details.

### ## Required dependencies

This type of dependency informs the `ComponentRuntime` that a given `Component` cannot function without a dependency.

When the dependency is missing during initialization, `ComponentRuntime` will throw a `MissingDependencyException`.

This type of dependency is useful for built-in components that are always present like `Context`, `FirebaseApp`, `FirebaseOptions`, [Executors]({{ site.baseurl }}{% link components/executors.md %}).

To declare a required dependency use one of the following in your `ComponentRegistrar`:

```
```java
// Required directly injected dependency
.add(Dependency.required(MyDep.class))
// Required lazily injected dependency
.add(Dependency.requiredProvider(MyOtherDep.class))
.factory( c -> new MyComponent(c.get(MyDep.class), c.getProvider(MyOtherDep.class)))
.build();
```
```

## ## Optional Dependencies

This type of dependencies is useful when your `Component` can operate normally when the dependency is not available, but can have enhanced functionality when present. e.g. `Firestore` can work without `Auth` but provides secure database access when `Auth` is present.

To declare an optional dependency use the following in your `ComponentRegistrar`:

```
```java
.add(Dependency.optionalProvider(MyDep.class))
.factory(c -> new MyComponent(c.getProvider(MyDep.class)))
.build();
```
```

The provider will return `null` if the dependency is not present in the app.

{: .warning }

When the app uses [Play's dynamic feature delivery](https://developer.android.com/guide/playcore/feature-delivery),

`provider.get()` will return your dependency when it becomes available. To support this use case, don't store references to the result of `provider.get()` calls.

See [Dynamic Module Support]({{ site.baseurl }}{% link components/dynamic\_modules.md %}) for details

{: .warning }

See Deferred dependencies if you your dependency has a callback based API

## ## Deferred Dependencies

Useful for optional dependencies which have a listener-style API, i.e. the dependent component registers a listener with the dependency and never calls it again (instead the dependency will call the registered listener). A good example is `Firestore`'s use of `Auth`, where `Firestore` registers a token change listener to get notified when a new token is available. The problem is that when `Firestore` initializes, `Auth` may not be present in the app, and is instead part of a dynamic module that can be loaded at runtime on demand.

To solve this problem, Components have a notion of a `Deferred` dependency. A deferred is defined as follows:

```
```java
public interface Deferred<T> {
    interface DeferredHandler<T> {
        @DeferredApi
        void handle(Provider<T> provider);
    }

    void whenAvailable(DeferredHandler<T> handler);
}
```
```

To use it a component needs to call `Dependency.deferred(SomeType.class)`:

```
```kotlin
class MyComponent(deferred: Deferred<SomeType>) {
    init {
        deferred.whenAvailable { someType ->
            someType.registerListener(myListener)
        }
    }
}
```
```

See [\[Dynamic Module Support\]](#)(`{{ site.baseurl }}{% link components/dynamic_modules.md %}`) for details

## ## Set Dependencies

The Components Framework allows registering components to be part of a set, such components are registered explicitly to be a part of a `Set<T>` as opposed to be a unique value of `T`:

```
```java
// Sdk 1
Component.intoSet(new SomeTypeImpl(), SomeType.class);
// Sdk 2
Component.intoSetBuilder(SomeType.class)
    .add(Dependency(SomeDep.class))
    .factory(c -> new SomeOtherImpl(c.get(SomeDep.class)))
    .build();
```
```

With the above setup each SDK contributes a value of `SomeType` into a `Set<SomeType>` which becomes available as a `Set` dependency.

To consume such a set the interested `Component` needs to declare a special kind of dependency in one of 2 ways:

- \* `Dependency.setOf(SomeType.class)`, a dependency of type `Set<SomeType>`.
- \* `Dependency.setOfProvider(SomeType.class)`, a dependency of type `Provider<Set<SomeType>>`. The advantage of this is that the `Set` is not initialized until the first call to `provider.get()` at which point all elements of the set will get initialized.

{: .warning }

Similar to optional `Provider` dependencies, where an optional dependency can become available at runtime due to [\[Play's dynamic feature delivery\]](https://developer.android.com/guide/playcore/feature-delivery)(<https://developer.android.com/guide/playcore/feature-delivery>), `Set` dependencies can change at runtime by new elements getting added to the set. So make sure to hold on to the original `Set` to be able to observe new values in it as they are added.

Example:

```
```kotlin
class MyClass(private val set1: Set<SomeType>, private val set2: Provider<Set<SomeOtherType>>)
```

```java
Component.builder(MyClass.class)
    .add(Dependency.setOf(SomeType.class))
    .add(Dependency.setOfProvider(SomeOtherType.class))
    .factory(c -> MyClass(c.setOf(SomeType.class), c.setOfProvider(SomeOtherType.class)))
    .build();
```
```

Apache License  
Version 2.0, January 2004  
<http://www.apache.org/licenses/>

## TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

### 1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or

Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work

or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
- (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
- (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work

by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.

Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[ ]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

## 1.38 jsoup 1.14.2

### 1.38.1 Available under license :

The MIT License

Copyright (c) 2009-2021 Jonathan Hedley <<https://jsoup.org/>>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 1.39 constraintlayout 2.1.4

## 1.39.1 Available under license :

No license file was found, but licenses were detected in source scan.

```
/*
 * Copyright (C) 2021 The Android Open Source Project
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
```

Found in path(s):

```
* /opt/cola/permits/1493901549_1677032761.60819/0/constraintlayout-2-1-4-sources-
jar/androidx/constraintlayout/motion/widget/TransitionAdapter.java
* /opt/cola/permits/1493901549_1677032761.60819/0/constraintlayout-2-1-4-sources-
jar/androidx/constraintlayout/motion/widget/OnSwipe.java
* /opt/cola/permits/1493901549_1677032761.60819/0/constraintlayout-2-1-4-sources-
jar/androidx/constraintlayout/widget/Group.java
* /opt/cola/permits/1493901549_1677032761.60819/0/constraintlayout-2-1-4-sources-
jar/androidx/constraintlayout/helper/widget/MotionPlaceholder.java
* /opt/cola/permits/1493901549_1677032761.60819/0/constraintlayout-2-1-4-sources-
jar/androidx/constraintlayout/helper/widget/Layer.java
* /opt/cola/permits/1493901549_1677032761.60819/0/constraintlayout-2-1-4-sources-
jar/androidx/constraintlayout/motion/widget/MotionHelper.java
* /opt/cola/permits/1493901549_1677032761.60819/0/constraintlayout-2-1-4-sources-
jar/androidx/constraintlayout/utils/widget/MockView.java
* /opt/cola/permits/1493901549_1677032761.60819/0/constraintlayout-2-1-4-sources-
jar/androidx/constraintlayout/widget/ConstraintHelper.java
* /opt/cola/permits/1493901549_1677032761.60819/0/constraintlayout-2-1-4-sources-
jar/androidx/constraintlayout/motion/utils/ViewState.java
* /opt/cola/permits/1493901549_1677032761.60819/0/constraintlayout-2-1-4-sources-
jar/androidx/constraintlayout/helper/widget/CircularFlow.java
```

No license file was found, but licenses were detected in source scan.

```
/*
 * Copyright (C) 2020 The Android Open Source Project
```

\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.  
\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/utils/ViewOscillator.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/helper/widget/MotionEvent.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/ReactiveGuide.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/ViewTransition.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/utils/ViewTimeCycle.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/utils/widget/MotionButton.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/utils/ViewSpline.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/ViewTransitionController.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/FloatLayout.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/utils/widget/MotionLabel.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/SharedValues.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/helper/widget/Carousel.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/MotionHelperInterface.java

No license file was found, but licenses were detected in source scan.

/\*  
\* Copyright (C) 2016 The Android Open Source Project  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/ConstraintSet.java  
No license file was found, but licenses were detected in source scan.

/\*  
\* Copyright (C) 2017 The Android Open Source Project  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.  
\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/ConstraintAttribute.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/Constraints.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/Placeholder.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/Barrier.java  
No license file was found, but licenses were detected in source scan.

/\*  
\* Copyright (C) 2019 The Android Open Source Project  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/TransitionBuilder.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/helper/widget/Flow.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/MotionInterpolator.java  
No license file was found, but licenses were detected in source scan.

/\*  
\* Copyright (C) 2015 The Android Open Source Project  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.  
\* You may obtain a copy of the License at  
\*  
\* <http://www.apache.org/licenses/LICENSE-2.0>  
\*  
\* Unless required by applicable law or agreed to in writing, software  
\* distributed under the License is distributed on an "AS IS" BASIS,  
\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
\* See the License for the specific language governing permissions and  
\* limitations under the License.  
\*/

Found in path(s):

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/ConstraintLayout.java  
\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/Guideline.java  
No license file was found, but licenses were detected in source scan.

/\*  
\* Copyright (C) 2018 The Android Open Source Project  
\*  
\* Licensed under the Apache License, Version 2.0 (the "License");  
\* you may not use this file except in compliance with the License.

- \* You may obtain a copy of the License at
- \*
- \* <http://www.apache.org/licenses/LICENSE-2.0>
- \*
- \* Unless required by applicable law or agreed to in writing, software
- \* distributed under the License is distributed on an "AS IS" BASIS,
- \* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
- \* See the License for the specific language governing permissions and
- \* limitations under the License.
- \*/

Found in path(s):

- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/Animatable.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/ConstraintsChangedListener.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/MotionLayout.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/KeyTimeCycle.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/KeyPosition.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/KeyTrigger.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/KeyFrames.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/MotionScene.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/Key.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/TouchResponse.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/ConstraintProperties.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/Debug.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/utils/widget/ImageFilterView.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/KeyCycle.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/utils/widget/MotionTellaes.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/StateSet.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/KeyPositionBase.java
- \* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/utils/StopLogic.java

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/MotionController.java

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/VirtualLayout.java

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/DesignTool.java

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/KeyAttributes.java

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/utils/widget/ImageFilterButton.java

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/MotionPaths.java

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/widget/ConstraintLayoutStates.java

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/CustomFloatAttributes.java

\* /opt/cola/permits/1493901549\_1677032761.60819/0/constraintlayout-2-1-4-sources-jar/androidx/constraintlayout/motion/widget/MotionConstrainedPoint.java

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: [www.cisco.com/go/trademarks](http://www.cisco.com/go/trademarks). Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

©2023 Cisco Systems, Inc. All rights reserved.