TECHNICAL SPECIFICATION

ISO/IEC TS 21425

First edition
2017-11

Programming languages — C++ Extensions for ranges

Langages de programmation — Extensions C++ pour les «ranges»



Reference number
ISO/IEC TS 21425:2017(E)

© ISO/IEC 2017



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2017, Published in Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

Foreword v

1 Scope 1

2 Normative references 1

3 Terms and definitions 1

4 General principles 2
4.1 Implementation compliance . . . 2
4.2 Namespaces, headers, and modifications to standard classes . . . 2

5 Statements 3
5.1 Iteration statements . . . 3

6 Library introduction 4
6.1 General . . . 4
6.2 Method of description (Informative) . . . 4
6.3 Library-wide requirements . . . 6

7 Concepts library 8
7.1 General . . . 8
7.2 Header `<experimental/ranges/concepts>` synopsis . . . 9
7.3 Core language concepts . . . 11
7.4 Comparison concepts . . . 16
7.5 Object concepts . . . 18
7.6 Callable concepts . . . 19

8 General utilities library 21
8.1 General . . . 21
8.2 Utility components . . . 21
8.3 Function objects . . . 22
8.4 Metaprogramming and type traits . . . 26
8.5 Tagged tuple-like types . . . 30

9 Iterators library 34
9.1 General . . . 34
9.2 Header `<experimental/ranges/iterator>` synopsis . . . 34
9.3 Iterator requirements . . . 42
9.4 Indirect callable requirements . . . 50
9.5 Common algorithm requirements . . . 52
9.6 Iterator primitives . . . 54
9.7 Iterator adaptors . . . 58
9.8 Stream iterators . . . 86

10 Ranges library 94
10.1 General . . . 94
10.2 decay_copy . . . 94
10.3 Header `<experimental/ranges/range>` synopsis . . . 94
10.4 Range access . . . 95
10.5 Range primitives . . . 97
10.6 Range requirements . . . 98

© ISO/IEC 2017 – All rights reserved

11 Algorithms library **101**
11.1 General . . . 101
11.2 Tag specifiers . . . 117
11.3 Non-modifying sequence operations . . . 118
11.4 Mutating sequence operations . . . 123
11.5 Sorting and related operations . . . 133

12 Numerics library **146**
12.1 Uniform random number generator requirements . . . 146

A Compatibility features **147**
A.1 General . . . 147
A.2 Rvalue range access . . . 147
A.3 Range-and-a-half algorithms . . . 147

B Acknowledgements **149**

C Compatibility **150**
C.1 C++ and Ranges . . . 150
C.2 Ranges and the Palo Alto TR (N3351) . . . 151

Bibliography **153**

Index **154**

Index of library names **155**

© ISO/IEC 2017 – All rights reserved

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*.

© ISO/IEC 2017 – All rights reserved

Programming languages — C++ Extensions for ranges

1 Scope [intro.scope]

1 This document describes extensions to the C++ Programming Language (2) that permit operations on ranges of data. These extensions include changes and additions to the existing library facilities as well as the extension of one core language facility. In particular, changes and extensions to the Standard Library include:

(1.1) — The formulation of the foundational and iterator concept requirements using the syntax of the Concepts TS (2).

(1.2) — Analogues of the Standard Library algorithms specified in terms of the new concepts.

(1.3) — The loosening of the algorithm constraints to permit the use of *sentinels* to denote the end of a range and corresponding changes to algorithm return types where necessary.

(1.4) — The addition of new concepts describing *range* and *view* abstractions; that is, objects with a begin iterator and an end sentinel.

(1.5) — New algorithm overloads that take range objects.

(1.6) — Support of *callable objects* (as opposed to *function objects*) passed as arguments to the algorithms.

(1.7) — The addition of optional *projection* arguments to the algorithms to permit on-the-fly data transformations.

(1.8) — Analogues of the iterator primitives and new primitives in support of the addition of sentinels to the library.

(1.9) — Constrained analogues of the standard iterator adaptors and stream iterators that satisfy the new iterator concepts.

(1.10) — New iterator adaptors (`counted_iterator` and `common_iterator`) and sentinels (`unreachable`).

2 Changes to the core language include:

(2.1) — the extension of the range-based `for` statement to support the new iterator range requirements (10.4).

3 This document does not specify constrained analogues of other parts of the Standard Library (e.g., the numeric algorithms), nor does it add range support to all the places that could benefit from it (e.g., the containers).

4 This document does not specify any new range views, actions, or facade or adaptor utilities; all are left as future work.

2 Normative references [intro.refs]

1 The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

(1.1) — ISO/IEC 14882:2014, *Programming Languages - C++*

(1.2) — ISO/IEC TS 19217:2015, *Programming Languages - C++ Extensions for Concepts*

ISO/IEC 14882:2014 is herein called the *C++ Standard* and ISO/IEC TS 19217:2015 is called the *Concepts TS*.

3 Terms and definitions [intro.defs]

For the purposes of this document, the terms and definitions given in ISO/IEC 14882:2014, ISO/IEC TS 19217:2015, and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at http://www.iso.org/obp

© ISO/IEC 2017 – All rights reserved

— IEC Electropedia: available at http://www.electropedia.org/

3.1 **[defns.expr.equiv]**

expression-equivalent

relationship that exists between two expressions `E1` and `E2` such that

— `E1` and `E2` have the same effects,

— `noexcept(E1) == noexcept(E2)`, and

— `E1` is a constant subexpression — an expression whose evaluation as subexpression of a conditional-expression `CE` (ISO/IEC 14882:2014 §5.16) would not prevent `CE` from being a core constant expression (ISO/IEC 14882:2014 §5.19) — if and only if `E2` is a constant subexpression

3.2 **[defns.projection]**

projection

⟨function object argument⟩ transformation which an algorithm applies before inspecting the values of elements

[*Example:*

```
std::pair<int, const char*> pairs[] = {{2, "foo"}, {1, "bar"}, {0, "baz"}};
ranges::sort(pairs, std::less<>{}, [](auto const& p) { return p.first; });
```

sorts the pairs in increasing order of their `first` members:

```
{{0, "baz"}, {1, "bar"}, {2, "foo"}}
```

— *end example*]

4 General principles [intro]

4.1 Implementation compliance [intro.compliance]

1 Conformance requirements for this specification are the same as those defined in ISO/IEC 14882:2014 §1.4. [*Note:* Conformance is defined in terms of the behavior of programs. — *end note*]

4.2 Namespaces, headers, and modifications to standard classes [intro.namespaces]

1 Since the extensions described in this document are experimental and not part of the C++ standard library, they should not be declared directly within namespace `std`. Unless otherwise specified, all components described in this document either:

(1.1) — modify an existing interface in the C++ Standard Library in-place,

(1.2) — are declared in namespace `std::experimental::ranges::v1`.

2 The International Standard, ISO/IEC 14882, together with ISO/IEC TS 19217:2015 (the Concepts TS), provide important context and specification for this document. In places, this document suggests changes to be made to components in namespace `std` in-place. In other places, entire chapters and sections are copied from ISO/IEC 14882 and modified so as to define similar but different components in namespace `std::experimental::ranges::v1`.

3 Instructions to modify or add paragraphs are written as explicit instructions. Modifications made to existing text from the International Standard use underlining to represent added text and ~~strikethrough~~ to represent deleted text.

4 This document assumes that the contents of the `std::experimental::ranges::v1` namespace will become a new constrained version of the C++ Standard Library that will be delivered alongside the existing unconstrained version.

5 Unless otherwise specified, references to other entities described in this document are assumed to be qualified with `std::experimental::ranges::`, and references to entities described in the International Standard are assumed to be qualified with `std::`.

6 New header names are prefixed with `experimental/ranges/`. Where the final element of a new header name is the same as an existing standard header name (e.g., `<experimental/ranges/algorithm>`), the new header shall include the standard header as if by

© ISO/IEC 2017 – All rights reserved