



**International
Standard**

ISO/IEC 14882

Programming languages — C++

Langages de programmation — C++

**Seventh
edition
2024-10**

iTeh Standards
(<https://standards.itih.ai>)
Document Preview

[ISO/IEC 14882:2024](https://standards.itih.ai/catalog/standards/iso/9e50ac69-a910-46ec-8c80-8648dda7b2db/iso-iec-14882-2024)

<https://standards.itih.ai/catalog/standards/iso/9e50ac69-a910-46ec-8c80-8648dda7b2db/iso-iec-14882-2024>

iTeh Standards
(<https://standards.itih.ai>)
Document Preview

[ISO/IEC 14882:2024](https://standards.itih.ai/catalog/standards/iso/9e50ac69-a910-46ec-8c80-8648dda7b2db/iso-iec-14882-2024)

<https://standards.itih.ai/catalog/standards/iso/9e50ac69-a910-46ec-8c80-8648dda7b2db/iso-iec-14882-2024>



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2024

All rights reserved. Unless otherwise specified, or required in the context of its implementation, 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
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Foreword	xi
Introduction	xiii
1 Scope	1
2 Normative references	2
3 Terms and definitions	3
4 General principles	9
4.1 Implementation compliance	9
4.2 Structure of this document	10
4.3 Syntax notation	11
5 Lexical conventions	12
5.1 Separate translation	12
5.2 Phases of translation	12
5.3 Character sets	13
5.4 Preprocessing tokens	16
5.5 Alternative tokens	17
5.6 Tokens	17
5.7 Comments	17
5.8 Header names	17
5.9 Preprocessing numbers	18
5.10 Identifiers	18
5.11 Keywords	19
5.12 Operators and punctuators	20
5.13 Literals	20
6 Basics	30
6.1 Preamble	30
6.2 Declarations and definitions	31
6.3 One-definition rule	32
6.4 Scope	37
6.5 Name lookup	42
6.6 Program and linkage	54
6.7 Memory and objects	58
6.8 Types	71
6.9 Program execution	78
7 Expressions	90
7.1 Preamble	90
7.2 Properties of expressions	90
7.3 Standard conversions	93
7.4 Usual arithmetic conversions	98
7.5 Primary expressions	99
7.6 Compound expressions	116
7.7 Constant expressions	148
8 Statements	157
8.1 Preamble	157
8.2 Label	158
8.3 Expression statement	158

8.4	Compound statement or block	158
8.5	Selection statements	158
8.6	Iteration statements	161
8.7	Jump statements	163
8.8	Declaration statement	164
8.9	Ambiguity resolution	165
9	Declarations	167
9.1	Preamble	167
9.2	Specifiers	169
9.3	Declarators	185
9.4	Initializers	201
9.5	Function definitions	217
9.6	Structured binding declarations	222
9.7	Enumerations	223
9.8	Namespaces	227
9.9	The <code>using</code> declaration	231
9.10	The <code>asm</code> declaration	235
9.11	Linkage specifications	236
9.12	Attributes	238
10	Modules	246
10.1	Module units and purviews	246
10.2	Export declaration	247
10.3	Import declaration	250
10.4	Global module fragment	251
10.5	Private module fragment	253
10.6	Instantiation context	254
10.7	Reachability	255
11	Classes	257
11.1	Preamble	257
11.2	Properties of classes	258
11.3	Class names	259
11.4	Class members	261
11.5	Unions	282
11.6	Local class declarations	284
11.7	Derived classes	285
11.8	Member access control	292
11.9	Initialization	301
11.10	Comparisons	313
12	Overloading	316
12.1	Preamble	316
12.2	Overload resolution	316
12.3	Address of an overload set	341
12.4	Overloaded operators	343
12.5	Built-in operators	346
12.6	User-defined literals	348
13	Templates	350
13.1	Preamble	350
13.2	Template parameters	351
13.3	Names of template specializations	355
13.4	Template arguments	357
13.5	Template constraints	362
13.6	Type equivalence	367
13.7	Template declarations	368
13.8	Name resolution	388

13.9	Template instantiation and specialization	401
13.10	Function template specializations	413
14	Exception handling	433
14.1	Preamble	433
14.2	Throwing an exception	434
14.3	Constructors and destructors	435
14.4	Handling an exception	436
14.5	Exception specifications	438
14.6	Special functions	440
15	Preprocessing directives	442
15.1	Preamble	442
15.2	Conditional inclusion	444
15.3	Source file inclusion	446
15.4	Module directive	447
15.5	Header unit importation	447
15.6	Macro replacement	449
15.7	Line control	454
15.8	Diagnostic directives	454
15.9	Pragma directive	455
15.10	Null directive	455
15.11	Predefined macro names	455
15.12	Pragma operator	458
16	Library introduction	459
16.1	General	459
16.2	The C standard library	460
16.3	Method of description	460
16.4	Library-wide requirements	467
17	Language support library	489
17.1	General	489
17.2	Common definitions	489
17.3	Implementation properties	493
17.4	Arithmetic types	504
17.5	Startup and termination	506
17.6	Dynamic memory management	507
17.7	Type identification	514
17.8	Source location	515
17.9	Exception handling	517
17.10	Initializer lists	521
17.11	Comparisons	522
17.12	Coroutines	530
17.13	Other runtime support	535
17.14	C headers	537
18	Concepts library	539
18.1	General	539
18.2	Equality preservation	539
18.3	Header <concepts> synopsis	540
18.4	Language-related concepts	542
18.5	Comparison concepts	547
18.6	Object concepts	550
18.7	Callable concepts	550
19	Diagnostics library	552
19.1	General	552
19.2	Exception classes	552

19.3	Assertions	555
19.4	Error numbers	555
19.5	System error support	557
19.6	Stacktrace	565
20	Memory management library	572
20.1	General	572
20.2	Memory	572
20.3	Smart pointers	590
20.4	Memory resources	617
20.5	Class template <code>scoped_allocator_adaptor</code>	626
21	Metaprogramming library	630
21.1	General	630
21.2	Compile-time integer sequences	630
21.3	Metaprogramming and type traits	630
21.4	Compile-time rational arithmetic	657
22	General utilities library	660
22.1	General	660
22.2	Utility components	660
22.3	Pairs	666
22.4	Tuples	672
22.5	Optional objects	686
22.6	Variants	699
22.7	Storage for any type	710
22.8	Expected objects	715
22.9	Bitsets	736
22.10	Function objects	742
22.11	Class <code>type_index</code>	769
22.12	Execution policies	771
22.13	Primitive numeric conversions	772
22.14	Formatting	775
22.15	Bit manipulation	801
23	Strings library	806
23.1	General	806
23.2	Character traits	806
23.3	String view classes	811
23.4	String classes	821
23.5	Null-terminated sequence utilities	849
24	Containers library	854
24.1	General	854
24.2	Requirements	854
24.3	Sequence containers	889
24.4	Associative containers	920
24.5	Unordered associative containers	939
24.6	Container adaptors	963
24.7	Views	1009
25	Iterators library	1036
25.1	General	1036
25.2	Header <code><iterator></code> synopsis	1036
25.3	Iterator requirements	1044
25.4	Iterator primitives	1065
25.5	Iterator adaptors	1068
25.6	Stream iterators	1094
25.7	Range access	1100

26 Ranges library	1102
26.1 General	1102
26.2 Header <code><ranges></code> synopsis	1102
26.3 Range access	1111
26.4 Range requirements	1115
26.5 Range utilities	1118
26.6 Range factories	1126
26.7 Range adaptors	1139
26.8 Range generators	1246
27 Algorithms library	1252
27.1 General	1252
27.2 Algorithms requirements	1252
27.3 Parallel algorithms	1254
27.4 Header <code><algorithm></code> synopsis	1257
27.5 Algorithm result types	1295
27.6 Non-modifying sequence operations	1298
27.7 Mutating sequence operations	1313
27.8 Sorting and related operations	1330
27.9 Header <code><numeric></code> synopsis	1356
27.10 Generalized numeric operations	1360
27.11 Specialized <code><memory></code> algorithms	1369
27.12 C library algorithms	1375
28 Numerics library	1376
28.1 General	1376
28.2 Numeric type requirements	1376
28.3 The floating-point environment	1376
28.4 Complex numbers	1377
28.5 Random number generation	1384
28.6 Numeric arrays	1426
28.7 Mathematical functions for floating-point types	1445
28.8 Numbers	1457
29 Time library	1459
29.1 General	1459
29.2 Header <code><chrono></code> synopsis	1459
29.3 <i>Cpp17Clock</i> requirements	1473
29.4 Time-related traits	1474
29.5 Class template <code>duration</code>	1475
29.6 Class template <code>time_point</code>	1482
29.7 Clocks	1485
29.8 The civil calendar	1495
29.9 Class template <code>hh_mm_ss</code>	1525
29.10 12/24 hours functions	1527
29.11 Time zones	1527
29.12 Formatting	1540
29.13 Parsing	1544
29.14 Header <code><ctime></code> synopsis	1548
30 Localization library	1549
30.1 General	1549
30.2 Header <code><locale></code> synopsis	1549
30.3 Locales	1550
30.4 Standard <code>locale</code> categories	1556
30.5 C library locales	1588

31 Input/output library	1590
31.1 General	1590
31.2 Iostreams requirements	1590
31.3 Forward declarations	1591
31.4 Standard iostream objects	1593
31.5 Iostreams base classes	1595
31.6 Stream buffers	1610
31.7 Formatting and manipulators	1618
31.8 String-based streams	1645
31.9 Span-based streams	1659
31.10 File-based streams	1666
31.11 Synchronized output streams	1679
31.12 File systems	1683
31.13 C library files	1728
32 Regular expressions library	1732
32.1 General	1732
32.2 Requirements	1732
32.3 Header <code><regex></code> synopsis	1734
32.4 Namespace <code>std::regex_constants</code>	1738
32.5 Class <code>regex_error</code>	1741
32.6 Class template <code>regex_traits</code>	1741
32.7 Class template <code>basic_regex</code>	1744
32.8 Class template <code>sub_match</code>	1747
32.9 Class template <code>match_results</code>	1749
32.10 Regular expression algorithms	1754
32.11 Regular expression iterators	1758
32.12 Modified ECMAScript regular expression grammar	1764
33 Concurrency support library	1766
33.1 General	1766
33.2 Requirements	1766
33.3 Stop tokens	1769
33.4 Threads	1774
33.5 Atomic operations	1782
33.6 Mutual exclusion	1815
33.7 Condition variables	1833
33.8 Semaphore	1840
33.9 Coordination types	1842
33.10 Futures	1845
Annex A Grammar summary	1860
A.1 General	1860
A.2 Keywords	1860
A.3 Lexical conventions	1860
A.4 Basics	1865
A.5 Expressions	1865
A.6 Statements	1869
A.7 Declarations	1870
A.8 Modules	1876
A.9 Classes	1877
A.10 Overloading	1878
A.11 Templates	1878
A.12 Exception handling	1879
A.13 Preprocessing directives	1880
Annex B Implementation quantities	1882

Annex C Compatibility	1884
C.1 C++ and ISO/IEC 14882:2020	1884
C.2 C++ and ISO/IEC 14882:2017	1888
C.3 C++ and ISO/IEC 14882:2014	1895
C.4 C++ and ISO/IEC 14882:2011	1899
C.5 C++ and ISO/IEC 14882:2003	1901
C.6 C++ and C	1906
C.7 C standard library	1914
Annex D Compatibility features	1917
D.1 General	1917
D.2 Arithmetic conversion on enumerations	1917
D.3 Implicit capture of <code>*this</code> by reference	1917
D.4 Array comparisons	1917
D.5 Deprecated <code>volatile</code> types	1917
D.6 Redclaration of <code>static constexpr</code> data members	1918
D.7 Non-local use of TU-local entities	1918
D.8 Implicit declaration of copy functions	1918
D.9 Literal operator function declarations using an identifier	1919
D.10 <code>template</code> keyword before qualified names	1919
D.11 Requires paragraph	1919
D.12 <code>has_denorm</code> members in <code>numeric_limits</code>	1919
D.13 Deprecated C macros	1919
D.14 Relational operators	1919
D.15 <code>char*</code> streams	1920
D.16 Deprecated error numbers	1927
D.17 The default allocator	1928
D.18 Deprecated <code>polymorphic_allocator</code> member function	1928
D.19 Deprecated type traits	1928
D.20 Tuple	1929
D.21 Variant	1930
D.22 Deprecated <code>iterator</code> class template	1930
D.23 Deprecated <code>move_iterator</code> access	1930
D.24 Deprecated <code>shared_ptr</code> atomic access	1931
D.25 Deprecated <code>basic_string</code> capacity	1933
D.26 Deprecated standard code conversion facets	1933
D.27 Deprecated convenience conversion interfaces	1934
D.28 Deprecated locale category facets	1938
D.29 Deprecated filesystem path factory functions	1938
D.30 Deprecated atomic operations	1938
Annex E Conformance with UAX #31	1940
E.1 General	1940
E.2 R1 Default identifiers	1940
E.3 R2 Immutable identifiers	1940
E.4 R3 <code>Pattern_White_Space</code> and <code>Pattern_Syntax</code> characters	1940
E.5 R4 Equivalent normalized identifiers	1941
E.6 R5 Equivalent case-insensitive identifiers	1941
E.7 R6 Filtered normalized identifiers	1941
E.8 R7 Filtered case-insensitive identifiers	1941
E.9 R8 Hashtag identifiers	1941
Bibliography	1942
Cross-references	1943
Cross-references from ISO/IEC 14882:2020	1970
Index	1971

Index of grammar productions	2005
Index of library headers	2011
Index of library names	2013
Index of library concepts	2096
Index of implementation-defined behavior	2100

iTeh Standards
(<https://standards.itih.ai>)
Document Preview

[ISO/IEC 14882:2024](https://standards.itih.ai/catalog/standards/iso/9e50ac69-a910-46ec-8c80-8648dda7b2db/iso-iec-14882-2024)

<https://standards.itih.ai/catalog/standards/iso/9e50ac69-a910-46ec-8c80-8648dda7b2db/iso-iec-14882-2024>

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.

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 <https://www.iso.org/directives> or https://www.iec.ch/members_experts/refdocs).

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at <https://www.iso.org/patents> and <https://patents.iec.ch>. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

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

For an explanation of 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 <https://www.iso.org/iso/foreword.html>. In the IEC, see <https://www.iec.ch/understanding-standards>.

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

This seventh edition cancels and replaces the sixth edition (ISO/IEC 14882:2020), which has been technically revised.

The main changes are as follows:

- improved support for Unicode;
- improved support for programming with constant expressions and constant evaluation;
- addition of a new way to declare non-static member functions with an “explicit `this` parameter”;
- addition of support for `#elifdef` and `#elifndef` preprocessing directives;
- change of overloaded `operator []` to allow multiple parameters;
- change of lifetime rules in range-based for loops;
- addition of a new “decay-copying” declaration “`auto(x)`”;
- support for extended floating-point types;
- addition of facilities for explicit lifetime management;
- addition of facilities for expressing assumptions;
- addition of standard library modules;
- addition of new standard library container and view types;
- addition of new standard library algorithms;
- addition of a generator type for use with coroutines;
- addition of an “expected” type for error handling;
- addition of string formatting and printing facilities;

— technical corrections and enhancements of existing core language and library facilities.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at <https://www.iso.org/members.html> and <https://www.iec.ch/national-committees>.

iTeh Standards
(<https://standards.itih.ai>)
Document Preview

[ISO/IEC 14882:2024](https://standards.itih.ai/catalog/standards/iso/9e50ac69-a910-46ec-8c80-8648dda7b2db/iso-iec-14882-2024)

<https://standards.itih.ai/catalog/standards/iso/9e50ac69-a910-46ec-8c80-8648dda7b2db/iso-iec-14882-2024>

Introduction

Clauses and subclauses in this document are annotated with a so-called stable name, presented in square brackets next to the (sub)clause heading (such as “[lex.token]” for 5.6, “Tokens”). Stable names aid in the discussion and evolution of this document by serving as stable references to subclauses across editions that are unaffected by changes of subclause numbering.

The cross references at the end of the document can be used to associate the stable names with their corresponding subclause number and to look up their location.

The indexes at the end of the document can be used to look up certain related entities such as grammar productions, names used by the standard library, and language constructs with implementation-defined behavior.

Aspects of the language syntax of C++ are distinguished typographically by the use of *italic*, *sans-serif* type or `constant width` type to avoid ambiguities; see 4.3.

iTeh Standards (<https://standards.iteh.ai>) Document Preview

[ISO/IEC 14882:2024](https://standards.iteh.ai/catalog/standards/iso/9e50ac69-a910-46ec-8c80-8648dda7b2db/iso-iec-14882-2024)

<https://standards.iteh.ai/catalog/standards/iso/9e50ac69-a910-46ec-8c80-8648dda7b2db/iso-iec-14882-2024>

