



**International
Standard**

ISO/IEC 9899

**Information technology —
Programming languages — C**

Technologies de l'information — Langages de programmation — C

**Fifth edition
2024-10**



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	xii
Introduction	xiii
1 Scope	1
2 Normative references	2
3 Terms, definitions, and symbols	3
4 Conformance	9
5 Environment	11
5.1 Introduction	11
5.2 Conceptual models	11
5.2.1 Translation environment	11
5.2.2 Execution environments	12
5.3 Environmental considerations	19
5.3.1 Character sets	19
5.3.2 Multibyte characters	20
5.3.3 Character display semantics	21
5.3.4 Signals and interrupts	21
5.3.5 Environmental limits	21
6 Language	35
6.1 Notation	35
6.2 Concepts	35
6.2.1 Scopes of identifiers, type names, and compound literals	35
6.2.2 Linkages of identifiers	36
6.2.3 Name spaces of identifiers	37
6.2.4 Storage durations of objects	37
6.2.5 Types	38
6.2.6 Representations of types	42
6.2.7 Compatible type and composite type	43
6.2.8 Alignment of objects	45
6.2.9 Encodings	45
6.3 Conversions	46
6.3.1 Introduction	46
6.3.2 Arithmetic operands	46

6.3.3	Other operands	49
6.4	Lexical elements	52
6.4.1	General	52
6.4.2	Keywords	53
6.4.3	Identifiers	54
6.4.4	Universal character names	56
6.4.5	Constants	57
6.4.6	String literals	67
6.4.7	Punctuators	68
6.4.8	Header names	69
6.4.9	Preprocessing numbers	70
6.4.10	Comments	70
6.5	Expressions	72
6.5.1	General	72
6.5.2	Primary expressions	73
6.5.3	Postfix operators	74
6.5.4	Unary operators	81
6.5.5	Cast operators	83
6.5.6	Multiplicative operators	84
6.5.7	Additive operators	85
6.5.8	Bitwise shift operators	86
6.5.9	Relational operators	86
6.5.10	Equality operators	87
6.5.11	Bitwise AND operator	88
6.5.12	Bitwise exclusive OR operator	88
6.5.13	Bitwise inclusive OR operator	89
6.5.14	Logical AND operator	89
6.5.15	Logical OR operator	89
6.5.16	Conditional operator	90
6.5.17	Assignment operators	91
6.5.18	Comma operator	94
6.6	Constant expressions	95
6.7	Declarations	97
6.7.1	General	97
6.7.2	Storage-class specifiers	98
6.7.3	Type specifiers	103
6.7.4	Type qualifiers	120
6.7.5	Function specifiers	124
6.7.6	Alignment specifier	125
6.7.7	Declarators	126

6.7.8	Type names	132
6.7.9	Type definitions	133
6.7.10	Type inference	134
6.7.11	Initialization	136
6.7.12	Static assertions	142
6.7.13	Attributes	142
6.8	Statements and blocks	152
6.8.1	General	152
6.8.2	Labeled statements	153
6.8.3	Compound statement	153
6.8.4	Expression and null statements	153
6.8.5	Selection statements	154
6.8.6	Iteration statements	155
6.8.7	Jump statements	156
6.9	External definitions	159
6.9.1	General	159
6.9.2	Function definitions	159
6.9.3	External object definitions	161
6.10	Preprocessing directives	163
6.10.1	General	163
6.10.2	Conditional inclusion	165
6.10.3	Source file inclusion	169
6.10.4	Binary resource inclusion	171
6.10.5	Macro replacement	178
6.10.6	Line control	185
6.10.7	Diagnostic directives	186
6.10.8	Pragma directive	186
6.10.9	Null directive	187
6.10.10	Predefined macro names	187
6.10.11	Pragma operator	189
6.11	Future language directions	190
6.11.1	Floating types	190
6.11.2	Linkages of identifiers	190
6.11.3	External names	190
6.11.4	Character escape sequences	190
6.11.5	Storage-class specifiers	190
6.11.6	Pragma directives	190
6.11.7	Predefined macro names	190

7 Library 191

7.1	Introduction	191
7.1.1	Definitions of terms	191
7.1.2	Standard headers	191
7.1.3	Reserved identifiers	192
7.1.4	Use of library functions	193
7.2	Diagnostics <assert.h>	195
7.2.1	General	195
7.2.2	Program diagnostics	195
7.3	Complex arithmetic <complex.h>	196
7.3.1	Introduction	196
7.3.2	Conventions	196
7.3.3	Branch cuts	197
7.3.4	The CX_LIMITED_RANGE pragma	197
7.3.5	Trigonometric functions	197
7.3.6	Hyperbolic functions	199
7.3.7	Exponential and logarithmic functions	200
7.3.8	Power and absolute-value functions	201
7.3.9	Manipulation functions	202
7.4	Character handling <ctype.h>	205
7.4.1	General	205
7.4.2	Character classification functions	205
7.4.3	Character case mapping functions	207
7.5	Errors <errno.h>	209
7.6	Floating-point environment <fenv.h>	210
7.6.1	General	210
7.6.2	The FENV_ACCESS pragma	212
7.6.3	The FENV_ROUND pragma	213
7.6.4	The FENV_DEC_ROUND pragma	215
7.6.5	Floating-point exceptions	216
7.6.6	Rounding and other control modes	218
7.6.7	Environment	221
7.7	Characteristics of floating types <float.h>	223
7.8	Format conversion of integer types <inttypes.h>	224
7.8.1	General	224
7.8.2	Macros for format specifiers	224
7.8.3	Functions for greatest-width integer types	225
7.9	Alternative spellings <iso646.h>	227
7.10	Characteristics of integer types <limits.h>	228
7.11	Localization <locale.h>	229
7.11.1	General	229

7.11.2	The setlocale function	229
7.11.3	Numeric formatting convention inquiry	230
7.12	Mathematics <math.h>	235
7.12.1	General	235
7.12.2	Treatment of error conditions	238
7.12.3	The FP_CONTRACT pragma	239
7.12.4	Classification macros	239
7.12.5	Trigonometric functions	242
7.12.6	Hyperbolic functions	247
7.12.7	Exponential and logarithmic functions	249
7.12.8	Power and absolute-value functions	257
7.12.9	Error and gamma functions	260
7.12.10	Nearest integer functions	262
7.12.11	Remainder functions	266
7.12.12	Manipulation functions	268
7.12.13	Maximum, minimum, and positive difference functions	270
7.12.14	Fused multiply-add	275
7.12.15	Functions that round result to narrower type	275
7.12.16	Quantum and quantum exponent functions	277
7.12.17	Decimal re-encoding functions	279
7.12.18	Comparison macros	281
7.13	Non-local jumps <setjmp.h>	284
7.13.1	General	284
7.13.2	Save calling environment	284
7.13.3	Restore calling environment	284
7.14	Signal handling <signal.h>	286
7.14.1	General	286
7.14.2	Specify signal handling	286
7.14.3	Send signal	288
7.15	Alignment <stdalign.h>	289
7.16	Variable arguments <stdarg.h>	290
7.16.1	General	290
7.16.2	Variable argument list access macros	290
7.17	Atomics <stdatomic.h>	294
7.17.1	Introduction	294
7.17.2	Initialization	295
7.17.3	Order and consistency	296
7.17.4	Fences	298
7.17.5	Lock-free property	299
7.17.6	Atomic integer types	300

7.17.7	Operations on atomic types	301
7.17.8	Atomic flag type and operations	303
7.18	Bit and byte utilities <stdbit.h>	305
7.18.1	General	305
7.18.2	Endian	305
7.18.3	Count Leading Zeros	306
7.18.4	Count Leading Ones	306
7.18.5	Count Trailing Zeros	306
7.18.6	Count Trailing Ones	307
7.18.7	First Leading Zero	307
7.18.8	First Leading One	308
7.18.9	First Trailing Zero	308
7.18.10	First Trailing One	309
7.18.11	Count Zeros	310
7.18.12	Count Ones	310
7.18.13	Single-bit Check	310
7.18.14	Bit Width	311
7.18.15	Bit Floor	311
7.18.16	Bit Ceiling	312
7.19	Boolean type and values <stdbool.h>	313
7.20	Checked Integer Arithmetic <stdckdint.h>	314
7.20.1	General	314
7.20.2	Checked Integer Operation Type-generic Macros	314
7.21	Common definitions <stddef.h>	315
7.21.1	General	315
7.21.2	The unreachable macro	316
7.21.3	The nullptr_t type	317
7.22	Integer types <stdint.h>	318
7.22.1	General	318
7.22.2	Integer types	318
7.22.3	Widths of specified-width integer types	320
7.22.4	Width of other integer types	320
7.22.5	Macros for integer constants	321
7.22.6	Maximal and minimal values of integer types	321
7.23	Input/output <stdio.h>	322
7.23.1	Introduction	322
7.23.2	Streams	324
7.23.3	Files	325
7.23.4	Operations on files	326
7.23.5	File access functions	328

7.23.6	Formatted input/output functions	331
7.23.7	Character input/output functions	349
7.23.8	Direct input/output functions	352
7.23.9	File positioning functions	353
7.23.10	Error-handling functions	355
7.24	General utilities <stdlib.h>	357
7.24.1	General	357
7.24.2	Numeric conversion functions	357
7.24.3	Pseudo-random sequence generation functions	364
7.24.4	Memory management functions	365
7.24.5	Communication with the environment	368
7.24.6	Searching and sorting utilities	371
7.24.7	Integer arithmetic functions	372
7.24.8	Multibyte/wide character conversion functions	373
7.24.9	Multibyte/wide string conversion functions	374
7.24.10	Alignment of memory	375
7.25	_Noreturn <stdnoreturn.h>	377
7.26	String handling <string.h>	378
7.26.1	String function conventions	378
7.26.2	Copying functions	378
7.26.3	Concatenation functions	380
7.26.4	Comparison functions	381
7.26.5	Search functions	382
7.26.6	Miscellaneous functions	385
7.27	Type-generic math <tgmath.h>	387
7.28	Threads <threads.h>	392
7.28.1	Introduction	392
7.28.2	Initialization functions	393
7.28.3	Condition variable functions	393
7.28.4	Mutex functions	395
7.28.5	Thread functions	397
7.28.6	Thread-specific storage functions	399
7.29	Date and time <time.h>	402
7.29.1	Components of time	402
7.29.2	Time manipulation functions	403
7.29.3	Time conversion functions	406
7.30	Unicode utilities <uchar.h>	412
7.30.1	General	412
7.30.2	Restartable multibyte/wide character conversion functions	412
7.31	Extended multibyte and wide character utilities <wchar.h>	417

7.31.1	Introduction	417
7.31.2	Formatted wide character input/output functions	418
7.31.3	Wide character input/output functions	431
7.31.4	General wide string utilities	435
7.31.4.1	General	435
7.31.4.2	Wide string numeric conversion functions	435
7.31.4.3	Wide string copying functions	440
7.31.4.4	Wide string concatenation functions	441
7.31.4.5	Wide string comparison functions	441
7.31.4.6	Wide string search functions	443
7.31.4.7	Miscellaneous functions	446
7.31.5	Wide character time conversion functions	446
7.31.6	Extended multibyte/wide character conversion utilities	447
7.31.6.1	General	447
7.31.6.2	Single-byte/wide character conversion functions	447
7.31.6.3	Conversion state functions	448
7.31.6.4	Restartable multibyte/wide character conversion functions	448
7.31.6.5	Restartable multibyte/wide string conversion functions	450
7.32	Wide character classification and mapping utilities <wctype.h>	452
7.32.1	Introduction	452
7.32.2	Wide character classification utilities	452
7.32.2.1	General	452
7.32.2.2	Wide character classification functions	452
7.32.2.3	Extensible wide character classification functions	455
7.32.3	Wide character case mapping utilities	456
7.32.3.1	Wide character case mapping functions	456
7.32.3.2	Extensible wide character case mapping functions	456
7.33	Future library directions	458
7.33.1	General	458
7.33.2	Complex arithmetic <complex.h>	458
7.33.3	Character handling <ctype.h>	458
7.33.4	Errors <errno.h>	458
7.33.5	Floating-point environment <fenv.h>	458
7.33.6	Characteristics of floating types <float.h>	458
7.33.7	Format conversion of integer types <inttypes.h>	458
7.33.8	Localization <locale.h>	458
7.33.9	Mathematics <math.h>	458
7.33.10	Signal handling <signal.h>	459
7.33.11	Atomics <stdatomic.h>	459
7.33.12	Boolean type and values <stdbool.h>	459

7.33.13 Bit and byte utilities <stdbit.h>	459
7.33.14 Checked Arithmetic Functions <stdckdint.h>	459
7.33.15 Integer types <stdint.h>	459
7.33.16 Input/output <stdio.h>	459
7.33.17 General utilities <stdlib.h>	459
7.33.18 String handling <string.h>	460
7.33.19 Date and time <time.h>	460
7.33.20 Threads <threads.h>	460
7.33.21 Extended multibyte and wide character utilities <wchar.h>	460
7.33.22 Wide character classification and mapping utilities <wctype.h>	460
Annex A (informative) Language syntax summary	461
Annex B (informative) Library summary	476
Annex C (informative) Sequence points	516
Annex D (informative) Universal character names for identifiers	517
Annex E (informative) Implementation limits	519
Annex F (normative) ISO/IEC 60559 floating-point arithmetic	522
Annex G (normative) ISO/IEC 60559-compatible complex arithmetic	553
Annex H (normative) ISO/IEC 60559 interchange and extended types	564
Annex I (informative) Common warnings	597
Annex J (informative) Portability issues	598
Annex K (normative) Bounds-checking interfaces	636
Annex L (normative) Analyzability	684
Annex M (informative) Change History	685
Bibliography	691
Index	692

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 www.iso.org/directives or 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 www.iso.org/patents and 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 www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC1, Information technology, Subcommittee SC22, Programming languages, their environments and system software interfaces.

This fifth edition cancels and replaces the fourth edition (ISO/IEC 9899:2018), which has been technically revised. The main changes are contained in Annex M.

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 www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

With the introduction of new devices and extended character sets, new features could be added to future editions of this document. Subclauses in the language and library clauses warn implementers and programmers of usages which, though valid in themselves, could conflict with future additions.

Certain features are *obsolescent*, which means that they could be considered for withdrawal in future revisions of this document. They are retained because of their widespread use, but their use in new implementations (for implementation features) or new programs (for language [6.11] or library features [7.33]) is discouraged.

This document is divided into four major subdivisions:

- preliminary elements (Clauses 1–4);
- the characteristics of environments that translate and execute C programs (Clause 5);
- the language syntax, constraints, and semantics (Clause 6);
- the library facilities (Clause 7).

In any given subsequent clause or subclause, there are section delineations in **bold** to describe the semantics, restrictions, and behaviors of programs for this language and potentially the use of its library clauses in this document:

- **Syntax**
which pertains to the spelling and organization of the language and library;
- **Constraints**
which detail and enumerate various requirements for the correct interpretation of the language and library, typically during translation;
- **Semantics**
which explain the behavior of language features and similar constructs;
- **Description**
which explain the behavior of library usage and similar constructs;
- **Returns**
which describes the effects of constructs provided back to a user of the library;
- **Runtime-constraints**
which detail and enumerate various requirements that are expected to be checked and which shall not be violated, typically during execution;
- **Environmental limits**
which list limitations an implementation may impose on a library or language construct which might otherwise be unlimited;
- **Recommended practice**
which provides guidance and important considerations for implementers of this document.

Examples are provided to illustrate possible forms of the constructions described. Footnotes are provided to emphasize consequences of the rules described in that subclause or elsewhere in this document. References are used to refer to other related subclauses. Recommendations are provided to give advice or guidance to implementers. Annexes define optional features, provide additional

information and summarize the information contained in this document. A bibliography lists documents that were referred to during the preparation of this document.

The language clause (Clause 6) is derived from “The C Reference Manual”[15].

The library clause (Clause 7) is based on the *1984 /usr/group Standard*[16].

Information technology — Programming languages — C

1. Scope

This document specifies the form and establishes the interpretation of programs written in the C programming language. It is designed to promote the portability of C programs among a variety of data-processing systems. It is intended for use by implementers and programmers. It specifies:

- the representation of C programs;
- the syntax and constraints of the C language;
- the semantic rules for interpreting C programs;
- the representation of input data to be processed by C programs;
- the representation of output data produced by C programs;
- the restrictions and limits imposed by a conforming implementation of C.

This document does not specify:

- the mechanism by which C programs are transformed for use by a data-processing system;
- the mechanism by which C programs are invoked for use by a data-processing system;
- the mechanism by which input data are transformed for use by a C program;
- the mechanism by which output data are transformed after being produced by a C program;
- the size or complexity of a program and its data that will exceed the capacity of any specific data-processing system or the capacity of a particular processor;
- all minimal requirements of a data-processing system that is capable of supporting a conforming implementation.

Annex J gives an overview of portability issues that a C program can encounter.

2. Normative references

The following documents are referred to in the text in such a way that some or all 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.

ISO/IEC 2382:2015, *Information technology — Vocabulary*.

ISO 4217, *Codes for the representation of currencies*.

ISO 8601 series, *Data elements and interchange formats — Information interchange — Representation of dates and times*.

ISO/IEC 10646, *Information technology — Universal Coded Character Set (UCS)*.

ISO/IEC 60559:2020, *Information technology — Microprocessor Systems — Floating-Point arithmetic*.

ISO 80000-2, *Quantities and units — Part 2: Mathematics*.

The Unicode Consortium. *Unicode Standard Annex, UAX #44, Unicode Character Database [online]*. Edited by Ken Whistler. Available at <https://www.unicode.org/reports/tr44>.

The Unicode Consortium. *The Unicode Standard, Derived Core Properties*. Available at <https://www.unicode.org/Public/UCD/latest/ucd/DerivedCoreProperties.txt>.