
**Information technology —
Programming languages — C**

Technologies de l'information — Langages de programmation — C

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

[ISO/IEC 9899:2018](https://standards.iteh.ai/catalog/standards/iso/1affe60f-8060-47f1-89c0-c6856c6eab89/iso-iec-9899-2018)

<https://standards.iteh.ai/catalog/standards/iso/1affe60f-8060-47f1-89c0-c6856c6eab89/iso-iec-9899-2018>



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

ISO/IEC 9899:2018

<https://standards.itih.ai/catalog/standards/iso/1affe60f-8060-47f1-89c0-c6856c6eab89/iso-iec-9899-2018>



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2018

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
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Foreword	xi
Introduction	xii
1 Scope	1
2 Normative references	2
3 Terms, definitions and symbols	3
4 Conformance	8
5 Environment	9
5.1 Conceptual models	9
5.1.1 Translation environment	9
5.1.2 Execution environments	10
5.2 Environmental considerations	17
5.2.1 Character sets	17
5.2.2 Character display semantics	19
5.2.3 Signals and interrupts	19
5.2.4 Environmental limits	19
6 Language	28
6.1 Notation	28
6.2 Concepts	28
6.2.1 Scopes of identifiers	28
6.2.2 Linkages of identifiers	29
6.2.3 Name spaces of identifiers	29
6.2.4 Storage durations of objects	30
6.2.5 Types	31
6.2.6 Representations of types	33
6.2.7 Compatible type and composite type	35
6.2.8 Alignment of objects	36
6.3 Conversions	37
6.3.1 Arithmetic operands	37
6.3.2 Other operands	40
6.4 Lexical elements	41
6.4.1 Keywords	42
6.4.2 Identifiers	43

6.4.3	Universal character names	44
6.4.4	Constants	45
6.4.5	String literals	50
6.4.6	Punctuators	52
6.4.7	Header names	53
6.4.8	Preprocessing numbers	53
6.4.9	Comments	54
6.5	Expressions	55
6.5.1	Primary expressions	56
6.5.2	Postfix operators	57
6.5.3	Unary operators	63
6.5.4	Cast operators	65
6.5.5	Multiplicative operators	66
6.5.6	Additive operators	66
6.5.7	Bitwise shift operators	68
6.5.8	Relational operators	68
6.5.9	Equality operators	69
6.5.10	Bitwise AND operator	70
6.5.11	Bitwise exclusive OR operator	70
6.5.12	Bitwise inclusive OR operator	70
6.5.13	Logical AND operator	71
6.5.14	Logical OR operator	71
6.5.15	Conditional operator	71
6.5.16	Assignment operators	72
6.5.17	Comma operator	75
6.6	Constant expressions	76
6.7	Declarations	78
6.7.1	Storage-class specifiers	79
6.7.2	Type specifiers	79
6.7.3	Type qualifiers	87
6.7.4	Function specifiers	90
6.7.5	Alignment specifier	92
6.7.6	Declarators	92
6.7.7	Type names	98
6.7.8	Type definitions	99
6.7.9	Initialization	100
6.7.10	Static assertions	105
6.8	Statements and blocks	106
6.8.1	Labeled statements	106
6.8.2	Compound statement	107

6.8.3	Expression and null statements	107
6.8.4	Selection statements	108
6.8.5	Iteration statements	109
6.8.6	Jump statements	110
6.9	External definitions	113
6.9.1	Function definitions	113
6.9.2	External object definitions	115
6.10	Preprocessing directives	117
6.10.1	Conditional inclusion	118
6.10.2	Source file inclusion	119
6.10.3	Macro replacement	121
6.10.4	Line control	126
6.10.5	Error directive	126
6.10.6	Pragma directive	127
6.10.7	Null directive	127
6.10.8	Predefined macro names	127
6.10.9	Pragma operator	129
6.11	Future language directions	130
6.11.1	Floating types	130
6.11.2	Linkages of identifiers	130
6.11.3	External names	130
6.11.4	Character escape sequences	130
6.11.5	Storage-class specifiers	130
6.11.6	Function declarators	130
6.11.7	Function definitions	130
6.11.8	Pragma directives	130
6.11.9	Predefined macro names	130
7	Library	131
7.1	Introduction	131
7.1.1	Definitions of terms	131
7.1.2	Standard headers	131
7.1.3	Reserved identifiers	132
7.1.4	Use of library functions	132
7.2	Diagnostics <assert.h>	135
7.2.1	Program diagnostics	135
7.3	Complex arithmetic <complex.h>	136
7.3.1	Introduction	136
7.3.2	Conventions	136
7.3.3	Branch cuts	136

7.3.4	The CX_LIMITED_RANGE pragma	137
7.3.5	Trigonometric functions	137
7.3.6	Hyperbolic functions	139
7.3.7	Exponential and logarithmic functions	140
7.3.8	Power and absolute-value functions	141
7.3.9	Manipulation functions	142
7.4	Character handling <ctype.h>	145
7.4.1	Character classification functions	145
7.4.2	Character case mapping functions	147
7.5	Errors <errno.h>	149
7.6	Floating-point environment <fenv.h>	150
7.6.1	The FENV_ACCESS pragma	151
7.6.2	Floating-point exceptions	152
7.6.3	Rounding	154
7.6.4	Environment	155
7.7	Characteristics of floating types <float.h>	157
7.8	Format conversion of integer types <inttypes.h>	158
7.8.1	Macros for format specifiers	158
7.8.2	Functions for greatest-width integer types	159
7.9	Alternative spellings <iso646.h>	161
7.10	Sizes of integer types <limits.h>	162
7.11	Localization <locale.h>	163
7.11.1	Locale control	163
7.11.2	Numeric formatting convention inquiry	164
7.12	Mathematics <math.h>	169
7.12.1	Treatment of error conditions	170
7.12.2	The FP_CONTRACT pragma	171
7.12.3	Classification macros	172
7.12.4	Trigonometric functions	173
7.12.5	Hyperbolic functions	175
7.12.6	Exponential and logarithmic functions	177
7.12.7	Power and absolute-value functions	180
7.12.8	Error and gamma functions	182
7.12.9	Nearest integer functions	183
7.12.10	Remainder functions	185
7.12.11	Manipulation functions	186
7.12.12	Maximum, minimum, and positive difference functions	187
7.12.13	Floating multiply-add	188
7.12.14	Comparison macros	189
7.13	Nonlocal jumps <setjmp.h>	191

7.13.1	Save calling environment	191
7.13.2	Restore calling environment	191
7.14	Signal handling <signal.h>	193
7.14.1	Specify signal handling	193
7.14.2	Send signal	194
7.15	Alignment <stdalign.h>	196
7.16	Variable arguments <stdarg.h>	197
7.16.1	Variable argument list access macros	197
7.17	Atomics <stdatomic.h>	200
7.17.1	Introduction	200
7.17.2	Initialization	201
7.17.3	Order and consistency	201
7.17.4	Fences	204
7.17.5	Lock-free property	205
7.17.6	Atomic integer types	205
7.17.7	Operations on atomic types	206
7.17.8	Atomic flag type and operations	208
7.18	Boolean type and values <stdbool.h>	210
7.19	Common definitions <stddef.h>	211
7.20	Integer types <stdint.h>	212
7.20.1	Integer types	212
7.20.2	Limits of specified-width integer types	213
7.20.3	Limits of other integer types	215
7.20.4	Macros for integer constants	216
7.21	Input/output <stdio.h>	217
7.21.1	Introduction	217
7.21.2	Streams	218
7.21.3	Files	219
7.21.4	Operations on files	221
7.21.5	File access functions	222
7.21.6	Formatted input/output functions	225
7.21.7	Character input/output functions	241
7.21.8	Direct input/output functions	244
7.21.9	File positioning functions	245
7.21.10	Error-handling functions	247
7.22	General utilities <stdlib.h>	249
7.22.1	Numeric conversion functions	249
7.22.2	Pseudo-random sequence generation functions	253
7.22.3	Memory management functions	254
7.22.4	Communication with the environment	256