

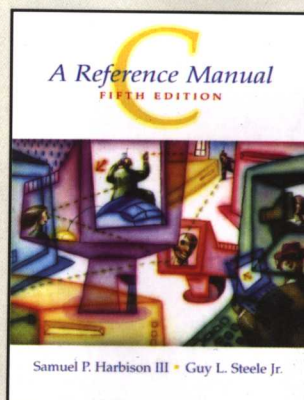


A Reference Manual
(Fifth Edition)

[美] Samuel P. Harbison III Guy L. Steele Jr. 著

C语言参考手册

(第5版)(英文版)



人民邮电出版社
POSTS & TELECOM PRESS



C 语言参考手册

(第5版)(英文版)

C A Reference Manual (Fifth Edition)

江苏工业学院图书馆
藏书章

著

人民邮电出版社

北京

图书在版编目 (CIP) 数据

C 语言参考手册: 第 5 版 / (美) 哈滨逊 (Harbison, S.P.) 等著.

—北京: 人民邮电出版社, 2007.7

ISBN 978-7-115-16188-8

I. C... II. 哈... III. C 语言—程序设计—技术手册—英文 IV. TP312-62

中国版本图书馆 CIP 数据核字 (2007) 第 062673 号

版 权 声 明

Original edition, entitled C A Reference Manual, Fifth Edition, 013089592x by Samuel P. Harbison III, Guy L. Steele Jr, published by Pearson Education, Inc, publishing as Prentice Hall, Copyright © 2002 by Prentice Hall. All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

China edition published by PEARSON EDUCATION ASIA LTD., and POSTS & TELECOMMUNICATIONS PRESS Copyright © 2007.

This edition is manufactured in the People's Republic of China, and is authorized for sale only in People's Republic of China excluding Hong Kong, Macau and Taiwan.

仅限于中华人民共和国境内 (不包括中国香港、澳门特别行政区和中国台湾地区) 销售。

本书封面贴有 Pearson Education (培生教育出版集团) 激光防伪标签。无标签者不得销售。

C 语言参考手册 (第 5 版) (英文版)

◆ 著 [美] Samuel P. Harbison III Guy L. Steele Jr.
责任编辑 刘映欣

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京顺义振华印刷厂印刷
新华书店总店北京发行所经销

◆ 开本: 800×1000 1/16

印张: 34.5

字数: 736 千字

印数: 1—3 000 册

2007 年 7 月第 1 版

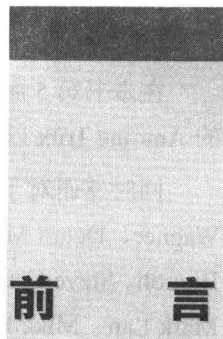
2007 年 7 月北京第 1 次印刷

著作权合同登记号 图字: 01-2003-0893 号

ISBN 978-7-115-16188-8/TP

定价: 55.00 元

读者服务热线: (010) 67132705 印装质量热线: (010) 67129223



前 言

本书是 C 语言的参考手册, 提供了 C 语言的基本概念和对运行库进行了完整和准确的描述, 并介绍了以正确性、可移植性和可维护性为根本出发点的良好的编程风格。

本书的预期读者是已经了解基本编程概念, 并且已经可以熟练使用 C 语言编程的人。为了保持参考手册的格式, 本书自下而上地介绍 C 语言的词法结构、预处理器、声明、类型、表达式、语句、函数和运行库。书中包括许多交叉引用, 以便于读者可以从任何地方入手。

这一版本完整地介绍了最新的国际 C 语言标准 ISO/IEC 9899:1999(C99), 明确指出语言本身和库函数中哪些特性是 C99 新增的, 并指出 C99 与原有 C89 标准的不同之处。本书是目前惟一适用于所有主流 C 语言版本的参考书: 包括传统 C 语言、1989 年 C 标准、1995 年对 C89 的修改与补充以及当前的 C99 标准。本书还介绍了标准 C 语言和标准 C++ 的原始 C 语言子集。尽管 C99 中增加了许多新内容, 但本书的章节结构没有做很大的修改, 这样就更便于熟悉旧版本的读者阅读。

本书的写作最初源于我们在 Tartan 公司的工作, 为从微机到大型机的一系列计算机开发 C 语言编译器系列。我们要求编译器的文档保持完整, 能够提供精确有用的错误诊断信息并能产生有效的目标代码。一个经过编译器正确编译的 C 语言程序, 应该能够在硬件存在差别的前提下在所有其他编译器中正确编译。

1984 年, C 语言已经非常普及, 但还没有精确地介绍 C 语言的书能够指导我们设计新的编译器。同样, 对于希望利用编译器更彻底地分析 C 语言程序的编程人员和客户来说, 当时的文档也不够精细, 因为当时人们习惯的方法已经不能够满足需要。本书特别注意影响程序清晰度、目标代码的有效性以及不同环境中程序移植性的语言特性。

Web 站点

欢迎访问本书的 Web 站点 <http://www.CAReferenceManual.com>, 可以在那里找到书中示例的代码、进一步的讨论、更正以及更多 C 语言资源的链接。

致谢

在本书第 5 版的编写过程中, 特别感谢原 NCITS J11 主席 Rex Jaeschke。芬兰赫尔辛基的 Antoine Trux 以及爱迪生设计集团创始人 Steve Adamczyk 的帮助。

同时感谢对于本书的以前版本给予过帮助的人 Jeffrey Esakov、Alan J. Filipski、Frank J. Wagner、Debra Martin、P. J. Plauger 以及 Steve Vinoski。其他提供过帮助的人还有 Aurelio Bignoli、Steve Clamage、Arthur Evans Jr., Roy J. Fuller、Morris M. Kessan、George V. Reilly、Mark Lan、Mike Hewett、Charles Fischer、Kevin Rodgers、Tom Gibb、David Lim、Stavros Macrakis、Steve Vegdahl、Christopher Vickery、Peter van der Linden 和 Dave Wilson。还要感谢 Michael Angus、Mady Bauer、Larry Breed、Sue Broughton、Alex Czajkowski、Robert Firth、David Gaffney、Steve Gorman、Dennis Hamilton、Chris Hanna、Ken Harrenstien、Rex Jaeschke、Don Lindsay、Tom MacDonald、Peter Nelson、Joe Newcomer、Kevin Nolish、David Notkin、Peter Plamondon、Roger Ray、Larry Rosler、David Spencer 以及 Barbara Steele。

本书最初的一些示例程序参考了以下作品中的算法:

- Beeler, Michael, Gosper, R. William, Schroepfel, Richard *HAKMEM* AI Memo 239 麻省理工大学人工智能实验室, 1972 年 2 月
- Bentley, Jon Louis *Writing Efficient Programs* Prentice-Hall, 1982
- Bentley, Jon Louis *Programming Pearls* 始于 1983 年 8 月的月刊 Communications of the ACM
- Kernighan, Brian W., Ritchie, Dennis M., *The C Programming Language* Prentice-Hall, 1978
- Knuth, Donald E., *The Art of Computer Programming* 卷 1-3 Addison-Wesley, 1968, 1969, 1973, 1981
- Sedgewick, Robert *Algorithms* Addison-Wesley, 1983

感谢这些作者的智慧。

还有一点需要说明, Guy Steele 先生因工作繁忙无法参与新版本的编写。虽然书中仍然处处体现出他对 C 语言精辟的分析, 但新版本中的任何新问题, 他不再负责。

《C 语言参考手册》一书已经出版了很多年, 感谢所有读者多年来对它的关爱。

Sam Harbison

宾西法尼亚 匹兹堡

harbison@CAREferenceManual.com

Contents

PART 1 The C Language	1
1 Introduction	3
1.1 The Evolution of C	3
1.2 Which Dialect of C Should You Use?	6
1.3 An Overview of C Programming	7
1.4 Conformance	8
1.5 Syntax Notation	9
2 Lexical Elements	11
2.1 Character Set	11
2.2 Comments	18
2.3 Tokens	20
2.4 Operators and Separators	20
2.5 Identifiers	21
2.6 Keywords	23
2.7 Constants	24
2.8 C++ Compatibility	38
2.9 On Character Sets, Repertoires, and Encodings	39
2.10 Exercises	41
3 The C Preprocessor	43
3.1 Preprocessor Commands	43
3.2 Preprocessor Lexical Conventions	44
3.3 Definition and Replacement	46
3.4 File Inclusion	59
3.5 Conditional Compilation	61
3.6 Explicit Line Numbering	66
3.7 Pragma Directive	67
3.8 Error Directive	69
3.9 C++ Compatibility	70
3.10 Exercises	71

4 Declarations**73**

- 4.1 Organization of Declarations 74
- 4.2 Terminology 75
- 4.3 Storage Class and Function Specifiers 83
- 4.4 Type Specifiers and Qualifiers 86
- 4.5 Declarators 95
- 4.6 Initializers 103
- 4.7 Implicit Declarations 113
- 4.8 External Names 113
- 4.9 C++ Compatibility 116
- 4.10 Exercises 119

5 Types**123**

- 5.1 Integer Types 124
- 5.2 Floating-Point Types 132
- 5.3 Pointer Types 136
- 5.4 Array Types 140
- 5.5 Enumerated Types 145
- 5.6 Structure Types 148
- 5.7 Union Types 160
- 5.8 Function Types 165
- 5.9 The Void Type 168
- 5.10 Typedef Names 168
- 5.11 Type Compatibility 172
- 5.12 Type Names and Abstract Declarators 176
- 5.13 C++ Compatibility 178
- 5.14 Exercises 179

6 Conversions and Representations**181**

- 6.1 Representations 181
- 6.2 Conversions 188
- 6.3 The Usual Conversions 194
- 6.4 C++ Compatibility 200
- 6.5 Exercises 201

7 Expressions**203**

- 7.1 Objects, Lvalues, and Designators 203
- 7.2 Expressions and Precedence 204
- 7.3 Primary Expressions 207
- 7.4 Postfix Expressions 210
- 7.5 Unary Expressions 219
- 7.6 Binary Operator Expressions 227
- 7.7 Logical Operator Expressions 243
- 7.8 Conditional Expressions 244
- 7.9 Assignment Expressions 246
- 7.10 Sequential Expressions 249

7.11	Constant Expressions	250
7.12	Order of Evaluation	253
7.13	Discarded Values	255
7.14	Optimization of Memory Accesses	256
7.15	C++ Compatibility	257
7.16	Exercises	258
8	Statements	259
8.1	General Syntactic Rules for Statements	260
8.2	Expression Statements	260
8.3	Labeled Statements	261
8.4	Compound Statements	262
8.5	Conditional Statements	264
8.6	Iterative Statements	266
8.7	Switch Statements	274
8.8	Break and Continue Statements	277
8.9	Return Statements	279
8.10	Goto Statements	280
8.11	Null Statements	281
8.12	C++ Compatibility	282
8.13	Exercises	282
9	Functions	285
9.1	Function Definitions	286
9.2	Function Prototypes	289
9.3	Formal Parameter Declarations	295
9.4	Adjustments to Parameter Types	298
9.5	Parameter-Passing Conventions	299
9.6	Agreement of Parameters	300
9.7	Function Return Types	301
9.8	Agreement of Return Types	302
9.9	The Main Program	303
9.10	Inline Functions	304
9.11	C++ Compatibility	306
9.12	Exercises	307
PART 2	The C Libraries	309
10	Introduction to the Libraries	311
10.1	Standard C Facilities	312
10.2	C++ Compatibility	313
10.3	Library Headers and Names	316
11	Standard Language Additions	325
11.1	NULL, ptrdiff_t, size_t, offsetof	325
11.2	EDOM, ERANGE, EILSEQ, errno, strerror, perror	327
11.3	bool, false, true	329

11.4	va_list, va_start, va_arg, va_end	329
11.5	Standard C Operator Macros	333
12	Character Processing	335
12.1	isalnum, isalpha, iscntrl, iswalnum, iswalpha, iswcntrl	336
12.2	iscsym, iscsymf	338
12.3	isdigit, isodigit, isxdigit, iswdigit, iswxdigit	338
12.4	isgraph, isprint, ispunct, iswgraph, iswprint, iswpunct	339
12.5	islower, isupper, iswlower, iswupper	340
12.6	isblank, isspace, iswhite, iswspace	341
12.7	toascii	341
12.8	toint	342
12.9	tolower, toupper, tolower, towupper	342
12.10	wctype_t, wctype, iswctype	343
12.11	wctrans_t, wctrans	344
13	String Processing	347
13.1	strcat, strncat, wscat, wcsncat	348
13.2	strcmp, strncmp, wstrcmp, wcsncmp	349
13.3	strcpy, strncpy, wscpy, wcsncpy	350
13.4	strlen, wcslen	351
13.5	strchr, strchr, wchr, wcschr	351
13.6	strspn, strspn, strpbrk, strpbrk, wcspn, wcspn, wcpbrk	352
13.7	strstr, strtok, wcsstr, wctok	354
13.8	strtod, strtod, strtold, strtol, strtoll, strtoul, strtoull	355
13.9	atof, atoi, atol, atoll	356
13.10	strcoll, strxfrm, wcscoll, wcsxfrm	356
14	Memory Functions	359
14.1	memchr, wmemchr	359
14.2	memcmp, wmemcmp	360
14.3	memcpy, memcpy, memmove, wmemcpy, wmemmove	361
14.4	memset, wmemset	362
15	Input/Output Facilities	363
15.1	FILE, EOF, wchar_t, wint_t, WEOF	365
15.2	fopen, fclose, fflush, freopen, fwide	366
15.3	setbuf, setvbuf	370
15.4	stdin, stdout, stderr	371
15.5	fseek, ftell, rewind, fgetpos, fsetpos	372
15.6	fgetc, fgetwc,getc, getwc, getchar, getwchar, ungetc, ungetwc	374
15.7	fgets, fgetws, gets	376
15.8	fscanf, fwscanf, scanf, wscanf, sscanf, swscanf	377
15.9	fputc, fputwc,putc, putwc, putchar, putwchar	385
15.10	fputs, fputws, puts	386
15.11	fprintf, printf, sprintf, snprintf, fwprintf, wprintf, swprintf	387

- 15.12 `v[x]printf, v[x]scanf` 401
- 15.13 `fread, fwrite` 402
- 15.14 `feof, ferror, clearerr` 404
- 15.15 `remove, rename` 404
- 15.16 `tmpfile, tmpnam, mktemp` 405

16 General Utilities**407**

- 16.1 `malloc, calloc, mlalloc, clalloc, free, cfree` 407
- 16.2 `rand, srand, RAND_MAX` 410
- 16.3 `atof, atoi, atol, atoll` 411
- 16.4 `strtod, strtod, strtold, strtol, strtoll, strtoul, strtoull` 412
- 16.5 `abort, atexit, exit, _Exit, EXIT_FAILURE, EXIT_SUCCESS` 414
- 16.6 `getenv` 415
- 16.7 `system` 416
- 16.8 `bsearch, qsort` 417
- 16.9 `abs, labs, llabs, div, ldiv, lldiv` 419
- 16.10 `mblen, mbtowc, wctomb` 420
- 16.11 `mbstowcs, wcstombs` 422

17 Mathematical Functions**425**

- 17.1 `abs, labs, llabs, div, ldiv, lldiv` 426
- 17.2 `fabs` 426
- 17.3 `ceil, floor, lrint, llrint, lround, llround, nearbyint, round, rint, trunc` 427
- 17.4 `fmod, remainder, remquo` 428
- 17.5 `frexp, ldexp, modf, scalbn` 429
- 17.6 `exp, exp2, expm1, ilogb, log, log10, log1p, log2, logb` 430
- 17.7 `cbrt, fma, hypot, pow, sqrt` 432
- 17.8 `rand, srand, RAND_MAX` 432
- 17.9 `cos, sin, tan, cosh, sinh, tanh` 433
- 17.10 `acos, asin, atan, atan2, acosh, asinh, atanh` 434
- 17.11 `fdim, fmax, fmin` 435
- 17.12 `Type-Generic Macros` 435
- 17.13 `erf, erfc, lgamma, tgamma` 439
- 17.14 `fpclassify, isfinite, isinf, isnan, isnormal, signbit` 440
- 17.15 `copysign, nan, nextafter, nexttoward` 441
- 17.16 `isgreater, isgreaterequal, isless, islessequal, islessgreater, isunordered` 442

18 Time and Date Functions**443**

- 18.1 `clock, clock_t, CLOCKS_PER_SEC, times` 443
- 18.2 `time, time_t` 445
- 18.3 `asctime, ctime` 445
- 18.4 `gmtime, localtime, mktime` 446
- 18.5 `difftime` 447

18.6	strftime, wcsftime	448
19	Control Functions	453
19.1	assert, NDEBUG	453
19.2	system, exec	454
19.3	exit, abort	454
19.4	setjmp, longjmp, jmp_buf	454
19.5	atexit	456
19.6	signal, raise, gsignal, ssignal, psignal	456
19.7	sleep, alarm	458
20	Locale	461
20.1	setlocale	461
20.2	localeconv	463
21	Extended Integer Types	467
21.1	General Rules	467
21.2	Exact-Size Integer Types	470
21.3	Least-Size Types of a Minimum Width	471
21.4	Fast Types of a Minimum Width	472
21.5	Pointer-Size and Maximum-Size Integer Types	473
21.6	Ranges of ptrdiff_t, size_t, wchar_t, wint_t, and sig_atomic_t	474
21.7	imaxabs, imaxdiv, imaxdiv_t	474
21.8	strtoimax, strtouimax	475
21.9	wcstoimax, wcstoumax	475
22	Floating-Point Environment	477
22.1	Overview	477
22.2	Floating-Point Environment	478
22.3	Floating-Point Exceptions	479
22.4	Floating-Point Rounding Modes	481
22.5	Floating-Point Expression Contraction	481
23	Complex Arithmetic	483
23.1	Complex Library Conventions	483
23.2	complex, _Complex_I, imaginary, _Imaginary_I, I	484
23.3	CX_LIMITED_RANGE	484
23.4	cacos, casin, catan, ccosh, csin, ctan	485
23.5	cacosh, casinh, catanh, ccosh, csinh, ctanh	486
23.6	cexp, clog, cabs, cpow, csqrt	487
23.7	carg, cimag, creal, conj, cproj	488
24	Wide and Multibyte Facilities	489
24.1	Basic Types and Macros	489
24.2	Conversions Between Wide and Multibyte Characters	490
24.3	Conversions Between Wide and Multibyte Strings	491
24.4	Conversions to Arithmetic Types	493
24.5	Input and Output Functions	493

Contents	7
24.6 String Functions	493
24.7 Date and Time Conversions	494
24.8 Wide-Character Classification and Mapping Functions	494
A The ASCII Character Set	497
B Syntax	499
C Answers to the Exercises	513
Index	521

List of Tables

Table 2-1	Graphic characters	12
Table 2-2	ISO trigraphs	15
Table 2-3	Operators and separators	21
Table 2-4	Keywords in C99	23
Table 2-5	Types of integer constants	27
Table 2-6	Assignment of types to integer constants	28
Table 2-7	Character escape codes	36
Table 2-8	Additional C++ keywords	39
Table 3-1	Preprocessor commands	44
Table 3-2	Predefined macros	52
Table 4-1	Identifier scopes	75
Table 4-2	Overloading classes	78
Table 4-3	Storage class specifiers	83
Table 4-4	Default storage class specifiers	84
Table 4-5	Function declarators	101
Table 4-6	Form of initializers	104
Table 4-7	Interpretation of top-level declarations	115
Table 5-1	C types and categories	124
Table 5-2	Values defined in <code>limits.h</code>	127
Table 5-3	Values defined in <code>float.h</code>	134
Table 5-4	IEEE floating-point characteristics	135
Table 6-1	Memory models on early PCs	186
Table 6-2	Permitted casting conversions	194
Table 6-3	Allowable assignment conversions	195
Table 6-4	Conversion rank	196
Table 6-5	Usual unary conversions (choose first that applies)	197
Table 6-6	Usual binary conversions (choose first that applies)	199
Table 7-1	Nonarray expressions that can be lvalues	204
Table 7-2	Operators requiring lvalue operands	204
Table 7-3	C operators in order of precedence	205
Table 7-4	Binary operator expressions	227
Table 7-5	Conditional expression 2nd and 3rd operands (pre-Standard)	245
Table 7-6	Conditional expression 2nd and 3rd operands (Standard C)	245
Table 7-7	Assignment operands	247

Table 7-8	Operand types for compound assignment expressions	249
Table 12-1	Property names for wctype	344
Table 15-1	Type specifications for fopen and freopen	368
Table 15-2	Properties of fopen modes	368
Table 15-3	Input conversions (scanf , fscanf , sscanf)	380
Table 15-4	Input conversions of the c specifier	382
Table 15-5	Input conversions of the s specifier	383
Table 15-6	Output conversion specifications	393
Table 15-7	Examples of the d conversion	394
Table 15-8	Examples of the u conversion	394
Table 15-9	Examples of the o conversion	395
Table 15-10	Examples of the x and X conversions	395
Table 15-11	Conversions of the c specifier	396
Table 15-12	Examples of the c conversion	396
Table 15-13	Conversions of the s specifier	397
Table 15-14	Examples of the s conversion	397
Table 15-15	Examples of the f conversion	398
Table 15-16	Examples of e and E conversions	399
Table 17-1	Type-generic macros	437
Table 18-1	Fields in struct tm type	447
Table 18-2	Formatting codes for strftime	449
Table 19-1	Standard signals	458
Table 20-1	Predefined setlocale categories	462
Table 20-2	lconv structure components	465
Table 20-3	Examples of formatted monetary quantities	465
Table 20-4	Examples of lconv structure contents	466
Table 21-1	Format control string macros for integer types (<i>N</i> = width of type in bits)	469
Table 23-1	Domain and range of complex trigonometric functions	485
Table 23-2	Domain and range of complex hyperbolic functions	486
Table 23-3	Domain and range of complex exponential and power	487
Table 23-4	Domain and range of miscellaneous complex functions	488
Table 24-1	Wide input/output functions	494
Table 24-2	Wide-string functions	495
Table 24-3	Wide-character functions	495

PART 1
The C Language

Introduction

Dennis Ritchie designed the C language at Bell Laboratories in the early 1970s, and its ancestry is traced from ALGOL 60 (1960), through Cambridge's CPL (1963), Martin Richards's BCPL (1967), and Ken Thompson's B language (1970) at Bell Labs. Although C is a general-purpose programming language, it has traditionally been used for systems programming. In particular, the popular UNIX operating system was originally written in C.

C's popularity is due to many factors. It is a small, efficient, yet powerful programming language with a rich run-time library. It provides precise control over the computer without a lot of hidden mechanisms. Since it has been standardized for over 10 years, programmers are comfortable with it. It is generally easy to write C programs that will be portable across different computing systems in different countries with different languages. Finally, there is a lot of legacy C code out there that is being modified and extended.

Starting in the late 1990s, C's popularity began to be eclipsed by its "big brother," C++. However, there is still a loyal following for the C language, and it continues to be popular where programmers do not need the features in C++ or where the overhead of C++ is not welcome.

C has withstood the test of time. It remains a language in which the experienced programmer can write quickly and well. Millions of lines of code testify to its usefulness.

1.1 THE EVOLUTION OF C

At the time we wrote the First Edition of this book in 1984, the C language was in widespread use, but there was no official standard or precise description of the language. The de facto standards were the C compilers being used. C became an international standard in 1989, was revised in 1994, and underwent a major revision in 1999.

Simply changing the definition of a language does not automatically alter the hundreds of millions of lines of C program code in the world. We have strived to keep this