

Programming in C

C程序设计

雍俊海 编著

清华大学出版社



Programming in C

C程序设计

雍俊海 编著

清华大学出版社
北京

内 容 简 介

本书讲解 C 语言程序设计知识及其编程方法,包括 C 语言的基础语法、结构化程序设计、静态数组、动态数组、字符串、结构体、共用体、枚举、函数、指针、单向链表、双向链表、类型别名、预处理命令、文件处理、编程规范、程序测试、自动测试、常用库函数与宏定义等内容。本书的内容不仅可以用在微软公司的 Windows 系列操作系统及其 VC 平台,而且也适用于 Linux 和 Unix 系列操作系统。本书的内容与章节编排以读者学习与认知过程为基础,与公司的实际需求相匹配。内容力求简洁,每章都附有习题,而且在附录中包含图、表、例程以及函数、运算符和宏的页码索引,采用特殊字体突出中心词,包含有注意事项、说明和小甜点等内容,希望使读者在轻松阅读的过程中迅速了解与掌握 C 语言程序设计的知识和方法,并应用到实践中。本书内容丰富易学,而且提供了大量的例程,既可以作为计算机专业和非计算机专业的基础教材以及 C 语言相关考试的辅导教材,也可以作为需要使用 C 语言的工程人员和科技工作者的自学参考书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

C 程序设计 / 雍俊海编著. —北京:清华大学出版社, 2017

ISBN 978-7-302-46675-8

I. ①C… II. ①雍… III. ①C 语言-程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字(2017)第 036028 号

责任编辑:魏江江 王冰飞

封面设计:傅瑞学

责任校对:李建庄

责任印制:宋 林

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社总机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 装 者:清华大学印刷厂

经 销:全国新华书店

开 本:185mm×260mm

印 张:32.25

字 数:785 千字

版 次:2017 年 6 月第 1 版

印 次:2017 年 6 月第 1 次印刷

印 数:1~2000

定 价:69.00 元

产品编号:039156-01

现代科学技术正在迅猛地发展着，计算机信息技术在其中发挥着巨大的作用，并已渗透到各行各业，推动着相关行业的迅速发展。合理利用计算机及其软件可以迅速提升人们生活与工作的效率。因此，如何尽快地掌握计算机知识，学好一门计算机语言，并用来解决人们在生活与工作中的实际问题，是一个有着迫切需求的问题。本书希望能在在这方面为读者“安上”智慧的翅膀，越过学好一门计算机语言的种种障碍，尽情享受学好一门计算机语言的乐趣。

C 语言是迄今为止人类发明的最为成功的计算机语言之一，应用非常广泛，同时也是很多其他计算机语言的基础。例如，C++语言是在 C 语言基础上发展起来的计算机语言，而 Java 语言和 C#语言则是在 C/C++语言的基础上发展起来的语言。无论如何发展与变革，C 语言本身依然不断展示出其巨大的自身优势，数十年来一直在开发商业软件所采用的计算机语言当中名列前茅，深受人们青睐。C 语言程序常常以短小精悍并且运行效率高著称。这实际上也体现出了 C 语言强大的表达能力和处理实际问题的能力。

无论国内还是国外，C 语言程序设计目前通常是计算机或软件专业学生学习的第一门计算机语言课程。这说明 **C 语言是一门相对比较容易入门的计算机语言**。它容易上手，而且可以迅速用来解决实际问题；具有很大的灵活性，可以支持结构化程序设计，可以用来培养严谨的编程思维习惯；具有很好的通用性，容易理解，可以构成学习其他计算机语言的基础。

我从 1991 年进入清华大学计算机系就开始学习 C 语言程序设计，积累了大量的笔记。2002 年我从美国留学回国并回到清华大学工作，那时便答应我的学生开始着手编写本书。因为工作的原因，编写的过程时断时续。也许因为自己过于追求完美，写起来非常缓慢，我从 VC (Visual C++，简称 VC) 6.0、VC 2003、VC 2005、VC 2008、VC 2010、VC 2012、VC 2015、Linux 的 gcc 和 Unix 的 gcc 等逐个平台进行实验，并反反复复地进行修改。在此期间，随着时间的流逝，两次出台了新的 C 语言国际标准，让我又改变了本书的写法。其实，因为工作的原因，C 语言与自己朝夕相处，要重现其完美是一件非常难的事。正所谓“亲近无伟人”。在描述与自己朝夕相处

的 C 语言的程序设计时，我一方面常常担心是否遗漏了什么；另一方面又担心过于烦琐，破坏了 C 语言的简洁之美。时间便在这反反复复之中流逝而去。好在最后终于成稿了，可以暂时松一口气了，也可以给学生一个交待了。

我希望本书能够给读者带来尽可能多的益处。对于学习而言，首先最重要的应当是对学习方法的引导。学习每门课程都有其内在的学习规律。顺应其规律，采用正确的学习方法，一般会产生良好的学习效果。学习首先应当是“学以致用”。为此，我常常利用各种机会调研软件公司对程序设计的实际需求。如果能够从应用出发进行学习，那么应当会提高学习的效率。另外，学习过程的关键应当是实践，课本是实践的一种辅助工具。为此，本书比较详细地讲解了在 VC 平台以及 Linux 和 Unix 操作系统下进行 C 语言程序设计实践的方法，努力降低进行 C 语言实践的难度，希望读者能够很快入门并进行 C 语言程序设计实践。在本书每章的后面都有习题。对于书中的习题，都没有提供答案，是因为我真诚希望这些习题能够给读者增加一些自主性思考和实践练习的机会。自主性思考意味着应当通过自己的思考去理解 C 语言并求解问题，同时应当不拘泥于某一种答案，即可以采用多种不同的方法求解相同的问题，从而快速掌握 C 语言及其程序设计方法。不过，本书提供了大量的例程及其详细的讲解，读者可以进行模仿。同时，希望读者能够经常总结实践过程的收获，享受其中的成就感，即使无法最终求解问题。

本书还讲解了 C 语言所有常用的库函数、运算符与宏，在一定程度上体现出了 C 语言程序设计手册或指南的特点。为了方便读者查找本书知识点和中心内容，通过加黑加粗加框的方式，强调各部分内容的中心词以及各个基本概念或定义的核心词，并提供了非常明显的注意事项、说明和小甜点等内容，而且在附录中添加了图、表、例程以及函数、运算符和宏的页码索引。此外，本书在鲁棒编程、高效编程和规范编程等方面也形成了一定的特色。

本书既可以作为计算机专业和非计算机专业的基础教材，也可以作为需要使用计算机的工程人员和科技工作者的自学参考书。本书在编写与出版的过程中得到了许多朋友的帮助，其中，读者、选修我所负责的课程的学生以及我所负责的清华大学计算机辅助设计、图形学与可视化研究所里的同事与学生，都起到了非常重要的作用。他们的建议和批评意见是本书发生变化的最重要的外在因素，并且他们当中的很多人也参与了本书的校对工作。本书也凝聚了他们的劳动结晶。这里一并对他们表示诚挚的谢意。真诚希望读者能够轻松并且愉悦地掌握 C 语言程序设计，也希望自己能做得更好。欢迎广大读者特别是讲授此课程的老师对本书进行批评和指正。真诚欢迎各种建设性意见。

雍俊海 于清华园

2017 年 3 月 3 日

本书约定

(1) 本书采用章、节和小节三级结构，其中小节是节的下一级结构。

(2) 在各种定义格式中，**斜体部分**表示某种**格式的模板**，在应用时需要进行替换。在格式模板中，中括号 “[] ” 表示其内部的选项不是必须的。

(3) 本书对一些字符和组合键的约定。本书采用字符 “↵” 表示**回车符**，即如果要求输入 “↵”，则表示按键盘上的回车键（或者称为 Enter 键）。为突出空格或者空格的个数，本书有时采用字符 “␣” 表示**空格**。例如，字符串 “a␣␣b” 分别由字符 ‘a’、空格、空格和 ‘b’ 组成。本书还会用到一些**组合键**。组合键通常由两个或两个以上的键组合而成。本书用加号 “+” 表示键的组合。例如，按组合键 Ctrl+s，表示同时按控制键（或称为 Control 键或 Ctrl 键）和字母 s 键。如果无法确保同时按这两个键，可以先按控制键不放，再按字母 s 键，然后同时放开这两个键。

(4) 单击鼠标键指的是按下鼠标键并迅速放开。双击鼠标键指的是快速地连续两次按下鼠标键并迅速放开。

图书资源支持

感谢您一直以来对清华版图书的支持和爱护。为了配合本书的使用,本书提供配套的素材,有需求的用户请到清华大学出版社主页(<http://www.tup.com.cn>)上查询和下载,也可以拨打电话或发送电子邮件咨询。

如果您在使用本书的过程中遇到了什么问题,或者有相关图书出版计划,也请您发邮件告诉我们,以便我们更好地为您服务。

我们的联系方式:

地 址: 北京海淀区双清路学研大厦 A 座 707

邮 编: 100084

电 话: 010-62770175-4604

资源下载: <http://www.tup.com.cn>

电子邮件: weijj@tup.tsinghua.edu.cn

QQ: 883604 (请写明您的单位和姓名)

用微信扫一扫右边的二维码,即可关注清华大学出版社公众号“书圈”。



扫一扫

资源下载、样书申请
新书推荐、技术交流

CONTENTS

目录

第 1 章 绪论	1
1.1 C 语言简介	1
1.2 开发 C 语言程序	4
1.2.1 第一个 C 语言例程	4
1.2.2 在 Microsoft Windows 下开发程序	8
1.2.3 在 Linux 或 Unix 下开发程序	12
1.3 小结	16
1.4 习题	17
第 2 章 数据和运算	18
2.1 标识符和关键字	18
2.2 数据类型	22
2.2.1 有符号整数系列类型和无符号整数系列类型	23
2.2.2 字符类型	26
2.2.3 枚举类型	27
2.2.4 浮点数类型	28
2.3 变量和字面常量	39
2.3.1 变量	39
2.3.2 有符号整数系列类型和无符号整数系列类型字面常量	43
2.3.3 字符类型字面常量	45
2.3.4 枚举类型字面常量	46
2.3.5 浮点数类型字面常量	47
2.4 数据的输入和输出	48
2.4.1 函数 printf	48
2.4.2 函数 scanf 和 scanf_s	54
2.4.3 字符输入函数 getchar 和字符输出函数 putchar	64

2.5	运算	66
2.5.1	算术运算	67
2.5.2	关系运算	75
2.5.3	逻辑运算	76
2.5.4	位运算	76
2.5.5	赋值类运算	80
2.5.6	条件运算	81
2.5.7	其他运算	82
2.6	小结	83
2.7	习题	84
第 3 章	控制结构	86
3.1	选择结构	86
3.1.1	if 语句和 if-else 语句	87
3.1.2	switch 语句	91
3.2	循环结构	94
3.2.1	for 语句	94
3.2.2	while 语句	96
3.2.3	do-while 语句	98
3.2.4	continue 语句	99
3.2.5	break 语句	101
3.3	小结	104
3.4	习题	105
第 4 章	结构化程序设计	108
4.1	函数基础	108
4.1.1	函数定义与调用	108
4.1.2	形式参数个数可变的函数	114
4.1.3	主函数 main	117
4.2	多个源程序文件	123
4.3	函数递归调用	130
4.4	结构化程序设计实现	142
4.5	小结	148
4.6	习题	148
第 5 章	静态数组	150
5.1	一维数组	151
5.2	多维数组	163
5.3	字符数组与字符串	172

5.4	小结	194
5.5	习题	195
第 6 章	结构体和共用体	196
6.1	结构体 (struct)	196
6.2	共用体 (union)	208
6.3	小结	215
6.4	习题	215
第 7 章	指针	216
7.1	指针类型与变量	216
7.2	动态数组	222
7.3	指针运算	229
7.4	函数返回值	230
7.5	单向链表和双向链表	241
7.6	函数指针与函数自动测试	270
7.7	小结	278
7.8	习题	278
第 8 章	关键字 typedef 和 const 以及预处理命令	280
8.1	类型别名定义 typedef	280
8.2	常量属性 const	285
8.3	预处理命令	288
8.3.1	宏定义#define 与取消宏定义#undef	289
8.3.2	条件编译	295
8.3.3	文件包含#include	298
8.4	小结	304
8.5	习题	304
第 9 章	文件处理	306
9.1	文件操作基本框架	306
9.2	以文本形式读写文件	316
9.3	以二进制数据流形式读写文件	324
9.4	文件整体信息与处理	327
9.5	错误处理	328
9.6	文件处理程序示例	330
9.7	小结	349

9.8 习题	349
第 10 章 编程规范、程序调试与测试	351
10.1 程序编写规范	352
10.1.1 命名规范	352
10.1.2 排版规范	358
10.1.3 语句	362
10.1.4 文件组织	369
10.2 程序编译与调试	376
10.2.1 程序编译与链接	376
10.2.2 设置断点与查看即时信息	382
10.2.3 断言 assert	389
10.2.4 编写调试日志文件程序	392
10.3 程序测试	396
10.4 小结	434
10.5 习题	435
第 11 章 其他常用函数和宏	437
11.1 数学库	437
11.1.1 指数与对数类函数	437
11.1.2 三角函数	446
11.1.3 反三角函数	449
11.1.4 双曲函数	452
11.1.5 其他数学类函数	454
11.2 标准库	458
11.2.1 伪随机数生成函数	458
11.2.2 程序退出函数以及注册函数	459
11.2.3 环境变量值的获取函数	462
11.2.4 查找与排序函数	463
11.2.5 整数绝对值和除法运算函数	467
11.3 时间获取与处理函数	469
11.3.1 在时间库中的时间获取函数	469
11.3.2 在时间库中的时间处理函数	471
11.3.3 在时间库中的时间格式输出函数	476
11.3.4 在 VC 平台下的高精度计时函数	481
11.4 小结	487
11.5 习题	487

附录 A 图的索引	490
附录 B 表的索引	492
附录 C 例程索引	494
附录 D 函数、运算符与宏索引	496
参考文献	502

C 语言是目前应用最广泛的计算机编程语言之一。一些公司的调研数据表明,在 2002 年到 2015 年期间,C 语言在所有计算机语言(包括 Java、C、C++、Basic、PHP、Perl、Python、C#和 JavaScript 等)中的占有率一直高居前两位,而且曾一度占据榜首,并且目前仍然具有很好的上升势头。目前,Unix 等操作系统的大部分代码是由 C 语言开发的。很多计算机语言(如 Fortran、Pascal、Python、Perl、LISP、Basic 和 LOGO)的编译器或解释器是采用 C 语言编写的。很多应用软件(如数据库软件 DBASE 和数学工具软件 MATLAB 等)也是采用 C 语言编写的。

1.1 C 语言简介

1946 年第一台电子计算机 ENIAC (Electronic Numerical Integrator And Computer, 电子数字积分计算机)在美国宾夕法尼亚州(Pennsylvania)诞生。从此,计算机以非常迅猛的速度发展。随着计算机的发展,计算机语言也在不断发展。最早的计算机语言称为第一代计算机语言,实际上就是机器语言。如图 1-1 (a) 所示,由机器语言编写的程序实际上就是直接由 0 和 1 两个数字组成的指令集。在图 1-1 (a) 中,由 0 和 1 组成的每一行数字序列实际上对应一个指令。采用机器语言编写的程序非常难以阅读。必须记住 0 和 1 组成的数字序列的含义及其格式,这样才能正确划分指令并理解其含义。另外,机器语言指令的格式与具体的计算机和操作系统密切相关,非常难以移植。

采用由 0 和 1 组成的数字序列编写程序确实非常不方便。到了 20 世纪 50 年代,开始出现了第二代计算机语言。最典型的第二代计算机语言是汇编语言。每条机器语言的指令基本上都对应一条汇编语言的指令。例如,在带有 Genuine Intel CPU (Central Processing Unit, 中央处理器)的个人计算机和 Microsoft Windows XP 操作系统下,图 1-1 (a) 所示的机器语言指令对应如图 1-1 (b) 所示的汇编语言指令。如图 1-1 (b) 所示,在汇编语言程序中,一些有含义的单词代替了由 0 和 1 组成的数字序列。例如, mov 表示赋值操作,

add 表示加法操作, imul 表示乘法操作。相对机器语言, 汇编语言确实提供了很大的编程便利。然而, 它仍然是一种低级的计算机语言, 其所需要编写的指令条数与机器语言的指令条数基本上相当。因此, 这时人们仍然只能以较低的效率编写程序。而且第二代计算机语言通常与计算机硬件和操作系统关系非常密切, 因此程序的可移植性较差。

<pre>10001011 01000101 11111100 00000011 01000101 11111000 10001001 01000101 11110100 10001011 01001101 11111000 00001111 10101111 01001101 11110100 10001001 01001101 11111100 10001011 01010101 11110100 10001001 01010101 11111000</pre>	<pre>mov eax,dword ptr [ebp-4] add eax,dword ptr [ebp-8] mov dword ptr [ebp-0Ch],eax mov ecx,dword ptr [ebp-8] imul ecx,dword ptr [ebp-0Ch] mov dword ptr [ebp-4],ecx mov edx,dword ptr [ebp-0Ch] mov dword ptr [ebp-8],edx</pre>	<pre>t = x+y; x = y*t; y = t;</pre>
(a) 机器语言程序片断	(b) 汇编语言程序片断	(c) C 语言程序片断

图 1-1 第一、二、三代计算机语言程序示意图

为了进一步降低编写程序的难度, 提高程序的编写效率, 到了 20 世纪 50 年代中后期, 开始出现了第三代计算机语言。第三代计算机语言属于高级语言。C、C++、Java 和 Basic 等目前常用的计算机语言都属于第三代计算机语言。相对前两代计算机语言, 采用高级语言编写的程序较容易被人理解。如图 1-1 所示, 一条 C 语言的语句可以对应多条汇编或机器语言指令。如图 1-1 (c) 所示的 C 语言语句所对应的汇编或机器语言指令在不同的计算机、不同的操作系统或不同的编译器下一般会稍有些不同。图 1-1 (a) 和图 1-1 (b) 只是显示其中的一种对应关系。

整个计算机学科正以非常迅猛的速度发展变化着, 与此相比, C 语言的发展变化相对缓慢一些。也许最初的 C 语言设计就已经相当完美了, 以至于目前难以对它进行各种优化或改进。面对快速变化的计算机世界, 学习 C 语言也许会有一种“一劳永逸”的感觉。表 1-1 给出了 C 语言发展的简要历史。

表 1-1 C 语言的发展简史

年份	事件
1960 年	Peter Naur 等人在巴黎国际会议上发表了 ALGOL 60 算法语言的报告, 它标志着 ALGOL 60 算法语言的诞生。ALGOL 60 算法语言是由欧美科学家联合开发的, 是 C 语言的原型。
1963 年	英国剑桥大学将 ALGOL 60 语言发展成为 CPL (Combined Programming Language) 语言。
1967 年	英国剑桥大学的 Martin Richards 对 CPL 语言作了简化, 推出了 BCPL (Basic Combined Programming Language) 语言。
1970 年	美国贝尔实验室的 Ken Thompson 以 BCPL 语言为基础, 进一步作简化, 设计出了 B 语言 (取自 BCPL 的第一个字母)。
1973 年	美国贝尔实验室的 Dennis M. Ritchie 在 B 语言的基础上设计出了 C 语言 (取自 BCPL 的第二个字母)。
1973 年	美国贝尔实验室的 Ken Thompson 和 Dennis M. Ritchie 两人合作把实现 Unix 操作系统本身 90% 以上的代码用 C 语言改写, 开发出 Unix 第 5 版。
1978 年	美国贝尔实验室的 Brian W. Kernighan 和 Dennis M. Ritchie 出版了著作《The C Programming Language》, 从而使 C 语言迅速得到推广。

续表

年份	事件
1983 年	美国国家标准化组织 (ANSI) 成立 X3J11 委员会制定 C 语言标准, 并于同年颁布了第一个 C 语言标准草案 (ANSI C 83)。在随后的几年里, 这个草案不断被讨论修订, 形成不同的 C 语言标准草案。
1987 年	美国国家标准化组织 (ANSI) 颁布了另一个 C 语言标准草案 (ANSI C 87)。
1989 年	美国国家标准化组织 (ANSI) 正式批准了 C 语言标准 ANSI C 89。
1990 年	国际标准化组织 (ISO) 采纳了 ANSI C 89 作为 C 语言的国际标准 ISO/IEC 9899:1990。
1999 年	C 语言标准 ANSI C 99 颁布。
2000 年	国际标准化组织 (ISO) 采纳了 ANSI C 99 作为 C 语言的国际标准 ISO/IEC 9899:1999。
2011 年	国际标准化组织 (ISO) 发布 C 语言国际标准 ISO/IEC 9899:2011。
2011 年	ANSI 采纳 ISO/IEC 9899:2011 作为 C 语言标准 ANSI C 11。

从表 1-1 中可以看出, C 语言是在不断发展变化着的。另外, 以 C 语言为起点产生了一些新的计算机语言。例如, C++语言是在 C 语言基础上发展起来的计算机语言, 而 Java 语言和 C#语言则是在 C/C++语言的基础上发展起来的语言。然而, C 语言本身并没有被这些新的计算机语言所取代。它目前仍然体现出巨大的自身优势, 依然是一种非常重要的计算机语言。人们通常不得不感叹“C 语言是迄今为止人类发明的最为成功的计算机语言之一”。通常认为 C 语言具有如下优点。

(1) **结构化的程序设计特点**: C 语言是一种面向过程的语言, 具有很好的结构化特性, 方便模块化设计和编程。结构化程序相对非结构化程序而言比较容易阅读和理解, 便于维护。

(2) **简单性**: 简单性是 C 语言标准制定的基本原则之一。C 语言的语法简洁, 编写出的程序通常比较紧凑。相对 Fortran 和 Pascal 而言, C 语言对程序代码的要求更为宽松。因此, 采用 C 语言可以更加容易表达程序设计的意图。相对 Java 和 C++等面向对象的计算机语言而言, 采用 C 语言编写的程序更为简洁, 而且通常更为短小。

(3) **高效性**: 通常认为 C 语言是一种相对高效的高级计算机语言。C 语言的简单性在一定程度上也保证了 C 语言的高效性。这种高效性体现在: ①采用 C 语言编写程序代码的效率通常较高; ②采用 C 语言编写的程序的运行效率通常较高, 一些统计资料表明采用 C 语言编写的程序的运行效率只比采用汇编语言编写的程序低约 10%。C 语言本身的限制少, 功能丰富。因此, 很多操作系统、编译系统和应用系统的大部分代码是由 C 语言编写的。

(4) **可移植性**: 在 C 语言自身的发展历程中, 让 C 语言程序具有较好的可移植性是 C 语言标准设计的一个重要目标。较好的可移植性意味着对计算机硬件和操作系统等具有较小的依赖性。C 语言可以广泛应用于不同的操作系统, 如 Microsoft Windows 以及 Linux 和 Unix 等。当然, C 语言程序的灵活性使得 C 语言程序的可移植性弱于 Java 语言。

如果只是不断强调 C 语言所具有的主要优点而不说明 C 语言的缺点, 似乎有些片面, 毕竟 C 语言仍然是一种在不断发展的计算机语言。通常认为 C 语言具有如下缺点。

(1) **代码的随意性**: C 语言编程的简单和灵活特性在一定程度上增加了代码出现错误的概率。相对宽松的语法使得 C 语言在编译时无法提前发现一些在程序运行时会发生的错误。如果不注意编程规范, 采用 C 语言很容易写出含糊难懂的代码。C 语言标准也指出有

些语句虽然符合语法要求,但其行为效果却是无法定义的。对于这样的语句,不同的 C 语言编译器有可能会进行不同的解析,从而产生不同的运行效果。因此,应当在程序代码中避免出现这些行为未定义的语句。后面的章节也会进一步具体介绍这样的内容。

(2) **内存管理**: 相对 Java 语言, C 语言拥有指针的概念,在内存使用上的语法限制少,缺乏内存安全检测和自动回收机制。在内存管理上无法通过面向对象语言的构造和析构函数机制形成一些统一的内存初始化和释放模式,从而需要更高的内存管理技巧。

(3) **并发特性**: C 语言是一种面向过程的语言,目前的 C 语言标准基本上没有考虑程序的并发运行特性。然而,多核计算机,即拥有多个 CPU 的计算机越来越常见。程序并发运行是未来计算机的重要发展方向。如何改进 C 语言标准,使其更好地支撑程序的并发运行特性,目前仍然是一个有待于解决的难题,也是国际标准化组织多年来一直想解决的难题。

目前人们正在努力去除 C 语言的缺点,同时希望尽量保持原有的优点,设计新的优良特性,并且兼容已有的 C 语言语法。这是一件非常艰巨而困难的事情。C 语言的优点和缺点就像是“双刃剑”,似乎常常是相伴而来。去掉它的缺点,有时不得不牺牲它的优点。这种取舍是非常困难的。改进和优化 C 语言语法的难度使得 C 语言新标准的制定需要较长的时间,也使得 C 语言的变化相对较为缓慢。当然,这在一定程度上降低了学习 C 语言和维护 C 语言程序的代价。尽管 C 语言拥有一些缺点,目前 C 语言仍然是很多操作系统和软件产品开发的的首选。

1.2 开发 C 语言程序

要学好任何一门计算机语言,都必须加强实践。学习和掌握 C 语言的基本知识、原理和方法,勇于尝试,多练习编写程序,努力利用程序求解各种实际问题,这基本上就是掌握和精通 C 语言的一般过程。**利用 C 语言求解实际问题的一般过程**是:首先进行需求分析,明确问题任务的详细需求,尤其是任务的输入、输出以及各种约束条件等;然后构造出基于 C 语言的求解模型,进行程序设计;接着将设计结果转化成为 C 语言语句并形成 C 语言程序;最后编译和运行程序,进行验证和调试。在这个过程中,**开发 C 语言程序的基本过程**可以用如图 1-2 所示的流程图表示。首先是根据需要创建和编辑项目文件或者 make 文件以及 C 语言源程序文件。项目文件或者 make 文件可以用来组织和管理 C 语言源程序文件,同时对编译和链接等进行设置。接着就可以进行编译(compile)和链接(link)。编译的结果将产生一些中间结果文件,这些中间结果文件通过链接生成最终的可执行文件。下面分别介绍在 Microsoft Windows 系列操作系统以及 Linux 或 Unix 系列操作系统下开发 C 语言程序的基本过程。

1.2.1 第一个 C 语言例程

本小节先给出第一个 C 语言例程的程序源代码,然后在后续的小节中给出在不同环境下如何编辑、编译和运行这个例程。

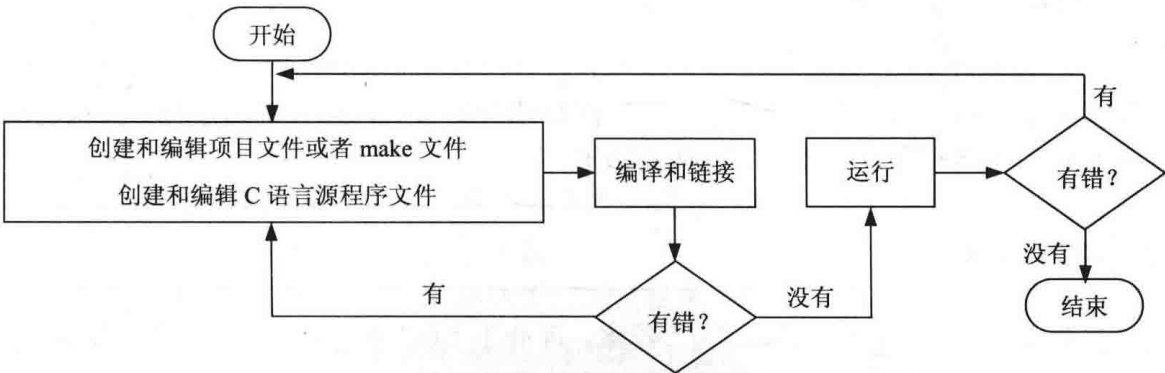


图 1-2 开发 C 语言程序的一般流程图

例程 1-1 简单招呼例程。

例程功能描述：该例程在控制台窗口中输出两行信息，其中第一行是“C 语言，您好！”，第二行是“我将成为优秀的 C 程序员！”。然后，程序等待用户按下键盘上的任意一个键。在接收到按键信息之后，程序运行结束。

这里直接给出**简单招呼例程**的源程序代码。它由一个源程序文件 C_Hello.c 组成。在源程序文件名 C_Hello.c 中，C_Hello 是文件的基本名，可以是任意的合法标识符；最后一个字母 c 是文件名的扩展名。C 语言源程序文件名的扩展名可以是小写字母 c，也可以是大写字母 C。下面给出源程序文件 C_Hello.c 的内容。

// 文件名：C_Hello.c；开发者：雍俊海	行号
#include <stdio.h>	// 1
#include <stdlib.h>	// 2
	// 3
int main(int argc, char* args[])	// 4
{	// 5
printf("C 语言，您好!\n");	// 6
printf("我将成为优秀的 C 程序员!\n");	// 7
system("pause"); // 暂停住控制台窗口	// 8
return 0; // 返回 0 表明程序运行成功	// 9
} // main 函数结束	// 10

这里对上面的源程序做初步解释，具体的说明将在以后的章节展开。在上面例程中，在每一行“//”之后的内容是程序的**注释**。C 语言的**注释**有两种，分别是行注释和块注释。**行注释**是以“//”引导的注释，即从“//”开始到行结束的内容都是注释。在早期的 C 语言程序代码中，不允许出现行注释。但是，自从 1999 年正式颁布 C 语言标准 ANSI C 99 以来，行注释就成为 C 语言的一个组成部分。

⚠️ 注意事项 ⚠️：在采用行注释时应当注意，这一行注释的末尾通常**不要以字符“\”结束**；否则，下一行代码也会被认为是行注释的一部分。

下面给出通过字符“\”对行注释进行续行的示例。