

面向21世纪

高职高专系列教材

C语言 程序设计

◎鲁 辉 主编

◎周岳山 审

 机械工业出版社
China Machine Press



面向 21 世纪高职高专系列教材

C 语言程序设计

鲁 辉 主编
周岳山 审



机械工业出版社

C语言是一种结构化的计算机语言，它既具有高级语言的特点，又具有低级语言的功能。本书是学习C语言程序设计的基础教程，采取循序渐进的内容安排，通俗易懂的讲解方式，并辅以大量的便于说明问题的例题，使得凡是学过一门高级语言的读者都能通过本书掌握C语言的基本内容，并应用它编写程序。本书内容精炼，结构合理。主要内容包括：C语言的基本概念、各种数据类型的使用技巧、各种表达式及语句结构、函数、指针、数组、结构体及共用体、文件等。本书可作为大专院校计算机专业的正式教材，也可供计算机培训班及其他自学者使用。

图书在版编目(CIP)数据

C语言程序设计/鲁辉主编. —北京：机械工业出版社，2002.8

面向21世纪高职高专系列教材

ISBN 7-111-10511-7

I. C... II. 鲁... III. C语言—程序设计—高等学校：技术学校—教材
IV. TP312

中国版本图书馆CIP数据核字(2002)第048758号

机械工业出版社（北京市百万庄大街22号 邮政编码100037）

策 划：胡毓坚

责任编辑：王冰飞

责任印制：闫 焱

北京京丰印刷厂印刷·新华书店北京发行所发行

2002年7月第1版·第1次印刷

1000mm×1400mm B5·9印张·349千字

0 001—5 000册

定价：22.00元

凡购本图书，如有缺页、倒页、脱页，由本社发行部调换

本社购书热线电话（010）68993821、68326677-2527

封面无防伪标均为盗版

面向 21 世纪高职高专 计算机专业系列教材编委会成员名单

顾问	曾玉崑	王文斌	陈瑞藻	李奇	凌林海
	林东				
主任委员	周智文				
副主任委员	周岳山(常务副主任)		詹红军	陈付贵	
	穆天保	赵佩华	黄甘洲	武文侠	吕何新
委员	郭曙光	王德年	刘瑞新	陈丽敏	孔令瑜
	李玲	鲁辉	陶书中	赵增敏	马伟
	孙心义	翟社平	廖常武	于恩普	王春红
	王娟萍	屈圭	汤新广	谢川	姜国忠
	汪赵强	董勇	梁国浚	张晓婷	
秘书长	胡毓坚				
副秘书长	陈丽敏(兼)				

出版说明

积极发展高职高专教育,完善职业教育体系,是我国职业教育改革和发展的一项重要任务。为了深化职业教育的改革,推进高职高专教育的发展,培养21世纪与我国现代化建设要求相适应的,并在生产、管理、服务第一线从事技术应用、经营管理、高新技术设备运作的高级职业技术应用型人才,尽快组织一批适应高职高专教学特色的教材,已成为各高职高专院校的迫切要求。为此,机械工业出版社与高职高专计算机专业、电子技术专业和机电专业教材编委会联合组织了全国40多所院校的骨干教师,共同研究开发了一批计算机专业、电子技术专业和机电专业的高职高专系列教材。

各编委会确立了“根据高职高专学生的培养目标,强化实践能力和创新意识的培养,反映现代职业教育思想、教育方法和教育手段,造就技术实用型人才为立足点”的编写原则。力求使教材体现“定位准确、注重能力、内容创新、结构合理和叙述通俗”的编写特色。

本套系列教材是由高职高专计算机专业、电子技术专业、机电专业教材编委会分别会同各院校第一线专业教师针对高职高专计算机、电子技术和机电各专业的教学现状和教材存在的问题开展研讨,尤其针对目前高职高专教学改革的新情况,分别拟定各专业的课程设置计划和教材选题计划。在教材的编制中,将教学改革力度比较大、内容新颖、有创新精神、比较适合教学、需要修编的教材以及院校急需、适合社会经济发展的新选题优先列入选题规划。在广泛征集意见及充分讨论的基础上,由各编委会确定每个选题的编写大纲和编审人员,实行主编负责制,编委会通过责任编辑和主审对教材进行质量监控。

担任本套教材编写的老师们都是来自各高职高专院校教育第一线的教师,他们以高度的责任感和使命感,经过近一年的努力,终于将本套教材呈现在广大读者面前。由于高职高专教育还处于起步阶段,加上我们的水平和经验有限,在教材的选题和编审中可能出现这样那样的问题,希望使用这套教材的教师和学生提出宝贵的意见和建议,以利我们今后不断改进,为我国的高职高专教育事业的繁荣而共同努力。

高职高专系列教材编委会
机械工业出版社

前 言

C语言是一种通用的程序设计语言。C语言以其数据类型丰富、功能完善、表达能力强、使用灵活、应用面广、目标程序效率高、可移植性好、能对硬件直接进行操作等特点吸引着众多的程序工作者。作为在系统软件和应用软件研制开发中不可缺少的工具,C语言的重要性日益突出。如著名的UNIX操作系统就是用C语言编写的。

本教材根据高职高专学生的培养目标,并结合全国计算机等级考试(二级)C语言程序设计考试大纲的要求,在内容上力求安排合理,突出重点,便于掌握。本教材的参考学时为70学时,共分为十二章。

第1章为程序设计与C程序,内容包括:C语言的产生过程及特点、程序与程序设计、算法、结构化程序设计和模块化结构等内容。第2章为数据类型与基本操作,主要描述了C语言的基本数据类型、运算符及表达式。第3、4、5章为顺序结构程序设计、选择结构程序设计和循环结构程序设计,主要介绍了C语言的输入输出函数、基本语句、条件语句、循环语句等。第6章介绍了数组数据类型。第7章介绍函数结构程序设计。第8章介绍指针数据类型,指针是C语言最具特色的标志之一。第9章和第11章描述了结构体、共用体、文件等构造数据类型及操作。第10章介绍了C语言中的位运算。为了便于学习,本教材配有大量的例题,并在每章结束配有试验和习题。同时,为了方便读者编程上机,本书在第12章还介绍了Turbo C集成开发环境的菜单命令。

本教材针对C语言入门容易精通难的特点,前面几章基础部分尽量作到概念清晰,由浅入深,使读者逐步掌握C语言的基本知识。本教材介绍的是C语言的最基本的部分,C语言在应用上的许多内容无法在这本基础教材中全部涉及,这有赖于读者将来在实践中逐步掌握。当然,这需要各位读者增加更多的计算机硬件知识。

本教材由鲁辉担任主编,周岳山负责审校工作。第1~6章、11章、12章由鲁辉编写,第7~10章由杨晔编写。由于编者水平有限,加之时间仓促,书中难免有疏漏及不足之处,敬请广大读者批评指正。

编者

2002年4月

目 录

出版说明	2.2.2 实型常量	19
前言	2.2.3 字符常量	19
第1章 程序设计与C程序	2.2.4 字符串常量	21
1.1 C语言的产生过程及特点	2.2.5 符号常量	21
1.1.1 C语言的产生过程	2.3 变量	22
1.1.2 C语言的特点	2.3.1 整型变量	22
1.2 程序与程序设计	2.3.2 实型变量	26
1.2.1 C程序与C程序格式	2.3.3 字符变量	31
1.2.2 程序设计的概念	2.3.4 变量的初始化	33
1.3 算法	2.4 表达式	34
1.3.1 算法的含义	2.4.1 表达式的概念	34
1.3.2 算法的构成要素	2.4.2 算术运算符与表达式	35
1.4 结构化程序设计和模块化结构	2.4.3 关系运算符与表达式	37
1.4.1 结构化程序	2.4.4 逻辑运算符与表达式	39
1.4.2 模块化结构	2.4.5 赋值运算符与赋值表 达式	42
1.4.3 C程序的调试	2.4.6 逗号运算符与逗号表 达式	47
1.5 C语言的元素及其构成	2.5 数据的类型转换	48
1.5.1 字符集	2.5.1 自动转换	48
1.5.2 标识符	2.5.2 强制转换	49
1.5.3 运算符	2.6 小结	50
1.5.4 常量	2.7 试验	51
1.5.5 变量	2.8 习题	51
1.5.6 注释符	第3章 顺序结构程序设计	54
1.6 小结	3.1 函数概述	54
1.7 试验	3.1.1 函数	54
1.8 习题	3.1.2 标准函数的调用	54
第2章 数据类型与基本操作	3.2 数据输出函数 printf	55
2.1 数据类型概述	3.2.1 printf函数的调用形式	55
2.2 常量	3.2.2 printf函数的简单用法	55
2.2.1 整型常量	3.2.3 printf函数的带格式输出	58

3.3 数据输入函数 scanf	65	4.8 习题	93
3.3.1 scanf 函数的调用形式	65	第 5 章 循环结构程序设计	95
3.3.2 scanf 函数中常用的格式 说明	66	5.1 while 语句构成的循环 结构	95
3.3.3 scanf 函数与输入缓冲区	70	5.1.1 while 语句的一般形式	95
3.4 getchar 函数与 putchar 函数	71	5.1.2 while 语句的特点	98
3.4.1 字符输入函数 getchar	71	5.2 do-while 语句构成的循环 结构	99
3.4.2 字符输出函数 putchar	72	5.2.1 do-while 语句的一般形式	99
3.5 基本语句	72	5.2.2 关于 do-while 循环的 说明	100
3.5.1 表达式语句	72	5.3 for 语句构成的循环结构	101
3.5.2 复合语句	73	5.3.1 for 语句的一般形式	101
3.6 应用举例	74	5.3.2 有关 for 语句的说明	102
3.7 小结	76	5.4 循环结构的嵌套	105
3.8 试验	77	5.5 循环辅助控制语句	107
3.9 习题	77	5.5.1 goto 语句	107
第 4 章 选择结构程序设计	79	5.5.2 break 语句	108
4.1 条件语句	79	5.5.3 continue 语句	109
4.1.1 if 语句	79	5.6 应用举例	110
4.1.2 嵌套的 if 语句	82	5.7 小结	114
4.2 条件表达式构成的选择 结构	86	5.8 试验	114
4.2.1 条件运算符	86	5.9 习题	115
4.2.2 条件表达式的运算功能	86	第 6 章 数组	117
4.2.3 条件运算符的优先级	86	6.1 一维数组	117
4.3 switch 语句与多路选择	87	6.1.1 一维数组的定义	117
4.3.1 switch 语句的一般形式	87	6.1.2 数组元素与数组的下标	118
4.3.2 switch 语句的执行过程	88	6.1.3 一维数组的初始化与 赋值	119
4.3.3 在 switch 语句体中使用 break 语句	89	6.1.4 数组元素的引用	122
4.4 语句标号和 goto 语句	90	6.2 二维数组	124
4.4.1 语句标号	90	6.2.1 二维数组和多维数组的 定义	124
4.4.2 goto 语句	90	6.2.2 二维数组与多维数组元素的 引用	126
4.5 应用举例	90	6.2.3 二维数组与多维数组的初	
4.6 小结	92		
4.7 试验	93		

始化	127	7.7.1 数组元素作为参数	158
6.2.4 应用举例	130	7.7.2 数组名作为参数	159
6.3 字符数组与字符串	133	7.8 字符串作为函数参数	161
6.3.1 字符串与字符串的存储	134	7.9 局部变量与全局变量	162
6.3.2 字符数组的初始化	135	7.9.1 局部变量	162
6.3.3 给一维字符数组赋字符 串值	136	7.9.2 全局变量	164
6.3.4 字符串的输入输出函数	137	7.10 变量的存储类型及其生存 期与作用域	165
6.3.5 字符串运算函数	140	7.10.1 变量的存储类型	165
6.3.6 二维字符数组	144	7.10.2 局部变量的生存期与作 用域	165
6.3.7 应用举例	146	7.10.3 全局变量的生存期与作 用域	167
6.4 小结	149	7.11 编译预处理	170
6.5 试验	149	7.11.1 宏替换	170
6.6 习题	149	7.11.2 宏包含	174
第7章 函数结构程序设计	151	7.11.3 条件编译	175
7.1 函数的概念	151	7.12 小结	176
7.1.1 函数	151	7.13 习题	176
7.1.2 函数的分类	151	第8章 指针	178
7.2 函数定义的一般形式	151	8.1 & 运算、* 运算及 sizeof 运算	178
7.3 函数的调用	153	8.2 指针变量的定义和引用	179
7.3.1 函数的两种调用方式	153	8.2.1 指针变量的定义	179
7.3.2 函数调用时的语法 要求	153	8.2.2 指针变量的初始化与 赋值	179
7.4 函数的说明	154	8.2.3 指针变量的引用	180
7.4.1 函数说明的形式	154	8.3 指针与函数参数	181
7.4.2 函数说明的位置	154	8.4 指针与数组	183
7.5 函数参数的传送方式	154	8.4.1 一维数组的指针表示 方法	183
7.5.1 形式参数与实在参数	154	8.4.2 二维数组的指针表示 方法	184
7.5.2 参数的传值传送方式	154	8.4.3 数组指针作为函数参数	186
7.5.3 参数的传址传送方式	155	8.5 指针与字符串	188
7.6 函数的返回值	156	8.5.1 字符串的存储形式	188
7.6.1 使用 return 语句传送返 回值	156		
7.6.2 使用地址参数传送返 回值	157		
7.7 数组作为函数参数	158		

8.5.2 字符串的另一种说明		9.5.1 动态存储分配和链表的概念	213
形式	188	9.5.2 用于动态存储分配的函数	214
8.6 指针数组	190	9.5.3 用包含指针项的结构体变量构成结点	214
8.7 指针与函数	192	9.5.4 应用举例	215
8.7.1 指针函数	192	9.6 共用体类型	217
8.7.2 函数指针	193	9.6.1 共用体的概念	217
8.8 指向指针的指针	195	9.6.2 共用体变量的定义和应用	217
8.9 命令行参数	196	9.6.3 应用举例	218
8.9.1 命令行参数的意义	196	9.7 枚举类型	219
8.9.2 使用命令行参数的方法	197	9.7.1 枚举类型的概念	219
8.10 小结	198	9.7.2 枚举类型变量的定义和应用	219
8.11 试验	199	9.7.3 应用举例	220
8.12 习题	199	9.8 用typedef 定义新类型名	221
第9章 结构体、共用体与枚举数据类型	202	9.9 小结	221
9.1 结构体类型	202	9.10 试验	222
9.1.1 结构体类型变量的定义	202	9.11 习题	223
9.1.2 结构体变量的初始化	203	第10章 位运算	224
9.1.3 结构体变量的引用	204	10.1 位运算概念	224
9.1.4 结构体变量的输入和输出	205	10.2 位运算符的运算功能	224
9.2 结构体数组	205	10.2.1 按位与运算符	224
9.2.1 结构体数组的定义方法	205	10.2.2 按位或运算符	224
9.2.2 结构体数组的初始化	206	10.2.3 按位异或运算符	224
9.2.3 结构体数组的引用	207	10.2.4 按位求反运算符	225
9.3 结构体与函数	208	10.2.5 左移运算符	225
9.3.1 结构体变量作为函数参数	208	10.2.6 右移运算符	225
9.3.2 返回结构体类型值的函数	210	10.3 位段	226
9.4 结构体与指针	211	10.3.1 位段的概念和定义方法	226
9.4.1 指向结构体变量的指针	211	10.3.2 位段的引用方法	226
9.4.2 指向结构体数组的指针	212	10.4 应用举例	227
9.4.3 指向结构体变量的指针作为函数参数	213		
9.5 结构体与链表	213		

10.5	小结	227	11.6.1	文件的定位	252
10.6	试验	228	11.6.2	随机读写	253
10.7	习题	228	11.7	文件操作的出错检测	257
第 11 章	文件	230	11.7.1	ferror 函数	257
11.1	文件的概念	230	11.7.2	clearerr 函数	258
11.2	缓冲文件系统和非缓冲 文件系统	231	11.8	小结	258
11.3	文件(FILE)类型指针	233	11.9	试验	258
11.4	文件的打开与关闭	234	11.10	习题	259
11.4.1	文件的打开	234	第 12 章	Turbo C 系统简介	260
11.4.2	文件的关闭	236	12.1	怎样使用 TC	260
11.5	文件的读写	237	12.2	Turbo C 集成开发环境的 菜单命令	261
11.5.1	输入和输出一个字符	237	附录	269
11.5.2	字符串读写函数	243	附录一:ASCII 字符编码表	269
11.5.3	格式化读写函数	246	附录二:运算符的优先级别和结合 方向	269
11.5.4	按“记录”的方式输入和 输出	248	附录三:库函数	270
11.6	文件的定位与随机 读写	252	参考文献	277

第 1 章 程序设计与 C 程序

1.1 C 语言的产生过程及特点

C 语言是一种通用计算机程序设计语言。20 世纪 60 年代,随着计算机科学的迅速发展,高级程序设计语言得到广泛的应用,然而,还没有一种可以用于书写操作系统和编译程序等系统程序的高级语言,人们不得不用汇编语言(机器语言)来书写,但汇编语言存在着不可移植、可读性差,研制软件效率不如高级语言等缺点,给编程带来很多不便。为此,人们对能用于系统程序设计的高级语言的开发就变得势在必行了,于是,20 世纪 70 年代初产生了一种能够用来研制各种系统程序的高级语言——C 语言。

1.1.1 C 语言的产生过程

C 语言的出现是与 UNIX 操作系统紧密联系在一起,C 语言本身也有一个产生过程,表 1-1 列出了 C 语言的发展历史。

表 1-1 C 语言的发展历史

语 言 名 称	设 计 者	年 份
CPL	C. Strachey	1968
BCPL	M. Richards	1969
B	K. Thompson	1970
C	D. M. Ritchie	1972

C 语言起源于 1968 年发表的 CPL(Combined Programming Language)语言。它的许多重要思想来源于 Martin Richards 在 1969 年研制的 BCPL(Basic Combined Programming Language)语言,以及以 BCPL 语言为基础而由 Ken Thompson 在 1970 年研制成的 B 语言。K. Thompson 用 B 语言写了第一个 UNIX 操作系统,用在 PDP-7 计算机(现已被淘汰)上。D. M. Ritchie 1972 年在 B 语言的基础上研制出 C 语言,并用 C 语言写了第一个在 PDP-11 计算机上实现的 UNIX 操作系统。UNIX 操作系统的巨大成功也伴随着 C 语言的巨大成功。

目前,从微型到大型计算机都配有 C 编译程序。不仅在装配 UNIX 操作系统的机器上,而且在非 UNIX 操作系统的机器上也配有多种 C 的编译程序。由于 C 语言本身具有许多特点,现在它已成为在微、小、大、巨型计算机上,从系统程序设计到工

程应用程序都能使用的一种高级程序设计语言。

1.1.2 C语言的特点

C语言以其简洁、灵活、表达能力强,产生的目标代码质量高,可移植性好为其基本特点而著称。C语言的特点可以从多方面来阐述,这里仅从使用者的角度加以讨论,其主要特点如下:

(1) 表达能力强且灵活。C语言是处于汇编语言和高级语言之间的一种记述性程序设计语言。C语言既具有面向硬件和系统,像汇编语言那样可以直接访问硬件的功能,又有高级语言面向用户、容易编程和维护、便于阅读和书写的优点。

(2) 程序结构清晰且紧凑。因为C语言通常由若干个函数组成,所以它是一种模块化程序设计语言。因此,它十分利于把整体程序分割成若干相对的功能模块。并且,它为程序模块间的相互调用以及数据传递提供了便利,这种模块化结构的程序不但清晰而且紧凑。

(3) 书写简单、易学。例如,C语言用“{”和“}”来代替Pascal语言中的begin和end作为复合语句括号,它的运算符也尽量缩写等。

(4) 目标程序的质量高。C语言提供了丰富的运算符集合,并且其中大多数运算符与一般机器指令相一致,可直接翻译成机器代码,因此,用它编写程序生成的代码质量高。实践证明,其他高级语言相对汇编语言的代码效率要低得多,而C语言的代码效率只比汇编语言低10%~20%。但C语言在描述问题时编程迅速、可读性好、表达能力强等优点是汇编语言无法相比的。

(5) 可移植性好。C语言的语句中,没有依存于硬件的输入/输出语句,程序的输入/输出功能是通过调用输入/输出函数实现的。而这些函数是由系统提供的独立于C语言的程序模块库,因此,C语言虽然具有直接访问硬件的功能,但C语言程序本身并不依存于机器硬件系统,从而便于在硬件结构不同的机种间实现程序的移植。

(6) C语言是一种结构化程序设计语言,特别适合大型程序的模块化设计。它提供了一整套循环、条件判断和转移语句,实现了对程序逻辑流程的有效控制,有利于结构化程序设计。

(7) C语言提供了丰富的数据类型。C语言具有五种基本的数据类型(字符型、整型、浮点单精度型、浮点双精度型、无值型)和多种构造数据类型(数组、结构、联合、枚举)以及导出类型。

(8) C语言允许程序员定义各种类型的变量指针和函数指针。指针是机器内存地址相关的说明项,因此指针是让程序员以相同于机器码的形式存取内存的数据。正确地使用指针可提高程序的效率。C语言还支持指针运算,允许程序员直接访问和操纵内存地址。

(9) C语言的预处理功能。C语言的预处理是一种正文处理,是编译之前对正文文件(源程序文件)的再安排。其中,用得最多的是程序的变量、代替函数调用的宏

(可较快运行)和基于某种特定条件的编译指令。

由于C语言具有上述众多特点,近年来迅速得到广泛普及和应用。C语言被称为“高级汇编语言”。特别是在微处理机和微型计算机的软件开发,以及各种软件工具的开发中,使用C语言的趋势日益增强,最近呈现出C语言有可能取代汇编语言的发展倾向。

1.2 程序与程序设计

1.2.1 C程序与C程序格式

1.C程序

当今,计算机已广泛应用于社会生活的各个领域,成为大众化的现代工具。但是,不熟悉计算机的人仍然把它想象得十分神秘。其实,计算机不过是一种具有内部存储能力、由程序自动控制的电子设备,当人们给出命令之后,它就按指令操作顺序自动进行。人们把这种可以连续执行的一条条指令的集合称为程序。可以说,程序就是人与机器进行对话的语言,也就是我们常说的“程序设计语言”。

目前,在社会上使用的程序设计语言有成百上千种,它们都被称为计算机的“高级语言”,如Basic、Pascal以及本书将介绍的C语言等。这些语言都是用接近人们习惯的自然语言和数学语言做为语言的表达形式,人们学习和操作起来感到十分方便。

但是,对于计算机本身来说,它并不能直接识别由高级语言编写的程序,它只能接受和处理由0和1的代码构成的二进制指令或数据。由于这种形式的指令是面向机器的,因此也称为“机器语言”。

我们把由高级语言编写的程序称为“源程序”,把由二进制代码表示的程序称为“目标程序”。如何把源程序转换成机器能够接受的目标程序,软件工作者编制了一系列软件,通过这些软件可以把用户按规定语法写出的语句一一翻译成二进制的指令。这种具有翻译功能的软件称为“编译程序”。每一种高级语言都有与它对应的编译程序。例如,C语言编译程序就是这样的一种软件,C语言编译程序的功能如图1-1所示。

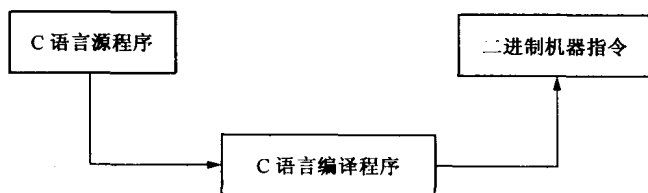


图 1-1 C语言编译程序的功能

我们用 C 语言编写的源程序,经过编译(Compile)最终都将转换成机器能直接识别的二进制机器指令形式的目标程序。由 C 语言构成的指令序列称 C 源程序,按 C 语言的语法编写 C 程序的过程,称 C 程序的代码编写。

C 语言源程序经过 C 语言编译程序编译之后生成一个后缀为 .obj 的二进制文件(称为目标文件)。最后还要由称为“连接程序”(Link)的软件,把此 obj 文件与 C 语言提供的各种库函数连接起来生成一个后缀为 .exe 的可执行文件。在 DOS 状态下,只需键入此文件的名字(而不必键入后缀 .exe),该执行文件就可运行。图 1-2 用以表示 C 程序的编辑、编译、连接、运行过程。

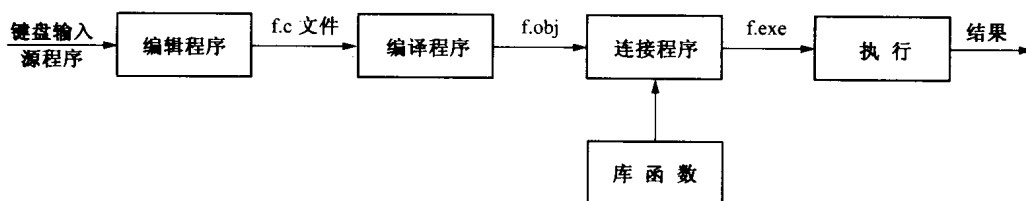


图 1-2 C 程序的执行全过程

2. C 语言的格式

任何一种程序设计语言都具有特定的语法规则和一定的表现形式。按照一定的格式和构成规则书写程序,不仅可以使程序设计人员和使用程序的人员易于理解,更重要的是,把程序输入到计算机时,计算机能够充分识别,并且正确执行它。

在研究 C 语言程序的格式前,先来看一个用 C 语言编写的求半径为 R 的圆面积的程序。

例 1.1

```
#include <stdio.h>
#define PI 3.1416
main( )
{
    int r;
    float S;      /* 变量说明 */
    scanf("%d",&r);      /* 调用输入函数 */
    S=PI*r*r;      /* 求圆面积 */
    printf("s= %f",s);   /* 调用输出函数 */
}
```

由上面程序可以看出,C 语言具有以下格式特点:

(1)main()是一个函数,而且是一个特殊的函数,每个程序必须有且只有一个名为 main 的函数。程序在 main() 函数(主函数)的开始处开始。main 名后面的圆括

号对()是必须的,圆括号对()中为参数表。main()可以有参数表,也可以没有参数表。这里为没有参数表的情况,但圆括号对()必须有,不能省去。

(2)C 语言程序习惯上使用英文小写字母书写,当然也可以用大写字母,但是大写字母在 C 语言中通常作为常量或其他特殊用途来使用。应当注意的是,C 语言对大小写是区分的,因此 count、Count、COUNT 在 C 语言中代表着不同的变量。

(3)C 语言程序是由一条条语句组成的,每个语句都具有规定的语法格式的特定的功能。上面程序中,scanf()语句是输入变量数值的函数调用语句;S=PI * r * r 是赋值语句;printf()是输出表达式数值的函数调用语句。

(4)C 语言程序不使用行号,这一点和 Basic 语言有区别。

(5)C 语言程序使用“;”号作为语句的结束符,它是语句的一部分。分号不是语句之间的分隔符。

(6)C 语言程序中,一个语句可以占用多行,一行也可以有多个语句。不过为了使程序易于阅读,一般一个语句占用一行。

(7)C 语言程序中使用大括号对“{”和“}”来表示程序的结构层次范围。一个完整的程序模块要用一对大括号括起来,以表示该程序模块的范围。应该注意的是,左大括号“{”和右大括号“}”应该对应使用。为了结构层次分明,书写程序时不同结构层次的语句,从不同的起始位置开始。同一结构层次中的语句,缩进相同个数的字符位置。同一结构层次中的花括号对“{”和“}”,亦缩进相同个数的字符位置。

(8)为了增加程序的可读性,可以适当地加一些注释行和空行。但是变量名、函数名和 C 语言保留字中间不能加入空格。除此之外注释行和空行可以任意设置,C 语言编译系统是不会理会这些注释行和空行的。

另外应当指出的是 C 语言程序的书写格式自由度较高,灵活性很强,有较大的随意性。但是为了避免程序的层次混乱不清,还是建议读者在平时的编程中,养成良好的编程习惯,使自己编写的程序便于别人阅读和理解。

1.2.2 程序设计的概念

什么是程序设计呢?最通俗的不严格的说法是:用计算机语言编写程序。那么什么是程序呢?著名科学家沃思提出了一个有名的公式:

算法 + 数据结构 = 程序

也就是说,要编写程序必须要研究如何组织数据,如何进行操作,数据是程序操作的对象,因此首先要确定对其施加操作的是什么样的数据,然后设计出对其操作的具体步骤,操作步骤就是算法。数据结构和算法是程序的两大要素。打个通俗的比方,一本菜谱介绍各种烹调方法,对每一种菜肴来说,需要说明用什么原料,然后介绍操作步骤,二者缺一不可。数据结构相当于原料,算法就是操作步骤,对不同的数据结构,算法是不同的。例如,对 10 个数排序,用数组处理和不用数组(只用简单变量)来处理,算法差别是很大的,选择合适的数据结构可以降低算法的复杂程度。

上述公式当时是针对较小规模的程序而提出来的,对于规模较大的程序而言,还要考虑如何组织程序设计。例如把一个大的任务先分解为若干较小的子任务,然后分别对每一个小任务(即模块)进行程序设计,这就降低了程序设计的复杂程度。这就要研究程序设计方法。近年来推广的结构化程序设计方法,目的是使程序设计工作规范化,改变过去一个时期中程序设计无章可循,把程序当作个人的“艺术品”从而降低程序的可读性的状况。结构化程序设计方法的要点包括:自顶向下;逐步细化;模块化。每一模块都由具有良好结构特征的基本结构组成。

如果要处理的是一个复杂庞大的任务,按照软件工程的要求,需进行系统分析、系统设计和程序设计,一个软件的生存周期经历以下几个阶段。

(1)问题定义与需求分析。

(2)总体设计(或称概要设计)。将一个大的任务划分为若干个子任务,即划分块。

(3)详细设计。根据给定子任务(模块)的要求,设计算法和数据结构。

(4)编写程序和单元测试。

(5)综合测试和确定运行。

(6)系统维护。

以上6个阶段称为“软件生成周期”,其中的(1)称为“软件定义时期”;(2)~(5)称为“软件开发时期”;(6)称为“软件维护时期”,一般所说的“程序设计”大体包括以上(3)、(4)两个阶段。

因此进行程序设计应该掌握程序设计方法,也就是要用结构化程序设计方法编写程序。

此外,一个程序必须是用一种计算机语言来编写的,要有必要的环境支持。例如,要用到操作系统提供的一些命令和系统软件。因此我们认为:

程序 = 算法 + 数据结构 + 程序设计方法 + 语言工具和环境

也就是说,一个合格的程序设计人员应当具备以上四个方面的知识。在这四个方面中,数据结构是一门专门的课程,包含丰富的内容,在本书中不能作专门的详细的论述,只能结合具体例题加以讨论。本书重点是讨论算法,根据算法编写程序,在本书中所有的程序都是按结构化程序设计方法编写的。本书介绍C语言的基本概念和使用方法,包括有关的环境支持,因此,学习了本书能够掌握以上四个方面的知识并用于程序设计。

程序设计的全过程应该包括:分析问题、划分模块、建立模型(物理模型和数学模型)、选择数据结构和算法、描述算法(例如画出流程图)、编程序、上机调试程序、分析运行结果、整理结果、写出程序说明书等文档。不要把程序设计简单地认为只是“编程序”而忽略其他方面的内容。