



计算机科学与技术系列丛书

C语言程序设计教程

C YUYAN CHENGXU SHEJI JIAOCHENG

■ 主编 胡金柱



教育部直属师范大学
华中师范大学出版社

JISUANJI

计算机科学与技术系列丛书

C 语言程序设计教程

主 编 胡金柱

副主编 贺 杰 胡 泉

编 写 (以姓氏笔画为序)

万 臣 王晶晶 刘智珺 胡 泉

贺 杰 谢 芳 彭 媛 董国刚

潘雪峰 魏妮妮

华中师范大学出版社

内 容 提 要

本书介绍了计算机高级语言 C 语言的基础知识和语法规则,以及应用 C 语言设计各种实用程序的方法。共分 11 章:第 1 章为 C 语言程序设计概论,第 2 章为 C 语言的数据类型、运算符与表达式,第 3 章为选择结构程序设计,第 4 章为循环结构程序设计,第 5 章为存储类型与数组,第 6 章为函数与编译预处理,第 7 章为指针基础,第 8 章为指针与函数,第 9 章为结构体与联合体,第 10 章为文件,第 11 章为 C 语言应用实例。

本书的特点是概念清楚,内容丰富,由浅入深,适用于不同需求的读者。每章配有练习题,便于教学和自学。

本书既可以作为高等院校计算机专业本科和专科学生初学计算机高级语言程序设计的教材,也可以作为其他各专业本科、专科计算机公共课教学的教材;既适用于作为参加计算机等级考试(二级 C 语言)的考前培训教材,也适用于计算机爱好者作自学教材。

新出图证(鄂)字 10 号

图书在版编目(CIP)数据

C 语言程序设计教程/胡金柱主编. —武汉:华中师范大学出版社,2007. 8

(计算机科学与技术系列丛书)

ISBN 978-7-5622-3583-5

I. C… II. 胡… III. C 语言—程序设计—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2007)第 080335 号

C 语言程序设计教程

主 编: 胡金柱

责任编辑: 裴媛媛©

责任校对: 罗 艺

封面设计: 罗明波

编辑室: 第二编辑室

电 话: 027-67867362

出版发行: 华中师范大学出版社

社 址: 武汉市武昌珞喻路 152 号

电 话: 027-67863040 027-67867076(发行部) 027-67861321(邮购)

传 真: 027-67863291

电子信箱: hscbs@public.wh.hb.cn

网 址: <http://www.ccupress.com>

经 销: 新华书店湖北发行所

印 刷 者: 武汉大学出版社印刷总厂

督 印: 章光琼

字 数: 470 千字

开 本: 787mm×1092mm 1/16

印 张: 18.5

版 次: 2007 年 8 月第 1 版

印 次: 2007 年 8 月第 1 次印刷

印 数: 1—5 100

定 价: 28.00 元

欢迎上网查询、购书

敬告读者: 欢迎举报盗版, 请打举报电话 027-67861321

前 言

C语言是一种通用的高级程序设计语言,它以丰富的数据类型、优美的表现能力、完善的功能特征和快速的编译环境等特点,吸引着众多的计算机工作者。C语言作为一门长盛不衰的高级程序设计语言,也深受广大学习计算机语言者的喜爱。

C语言已经成为全球程序员使用的公共编程语言,并由此产生了当前程序设计的两个主流语言 C++ 和 Java。它们都是建立在 C 语言的语法和基本结构基础上的,而且现在世界上的许多软件都是在 C 语言及其衍生的各种语言的基础上开发形成的。所以 C 语言不仅是学习 C++ 和 Java 语言的基础语言,也是学习其他计算机高级语言程序设计的基础语言。

目前绝大多数高等院校的计算机专业都将 C 语言作为培养学生程序设计能力的首选语言,绝大多数高等院校的非计算机专业的计算机公共课也都开设 C 语言程序设计,而且全国计算机等级考试二级考试中也把 C 语言程序设计作为首选语言。另外,目前许多单位和计算机公司招收计算机软件人员,都要考 C 语言程序设计。特别是面试,一般都会考核应聘者上机调试 C 语言程序的能力。许多高等院校计算机专业招收硕士研究生,其面试也是如此。

目前许多高等院校,无论是计算机专业还是非计算机专业,都将 C 语言程序设计作为第一门程序设计课程,一般都会安排在第 1 学期或者第 2 学期。由于在高级语言程序设计的课程中给定了具体的语言(多数为 C 语言),不少教师不自觉地将教学中的“程序设计”变成了“一种高级语言的程序编写”,过多地强调语言成分的语法和简单语义。经验表明,这种一开始就将学生带入实际为“一种具体的高级语言的程序编写”的做法容易对学生产生误导,使学生片面地认为“计算机就是编程序”,一开始就扎进程序设计的语言细节中去。这不仅使学生忽视了重要的理论、算法等问题,更重要的是,由于没能使学生在头脑中形成如何利用计算机进行“计算”的概念,还会使学生在遇到问题需要用程序进行求解时,“满脑袋都是语句,却不知道应该写哪个语句”。当需要到机器上实验时,会看到一些学生坐在机器旁,只会照着课本“敲”例题,“一分一毫”也不敢错。其主要原因是把“程序设计”课讲成了“一种具体的高级语言的程序编写”课。

所以,我们应把重点放在程序设计实践以及培养学生分析问题和解决问题的能力训练方面,首先必须使学生养成“程序设计”的习惯,初步掌握程序设计的基本方法和思想(如抽象的思维方法和自顶向下的设计思想)。在有了这个基础以后,才能引导他们用具体的高级语言进行“程序编写”。

对于学生,具备了真正的“程序设计”基础,不仅为今后顺利地更换“程序编写”的语言提供了支撑,也对学习后续的数据结构与算法课程、理解典型系统软件的设计、掌握有关的设计方法打下良好的基础,对学生今后在新环境或新技术下进行系统设计和实现也是最好的支持。

C 语言程序设计的上机实验,不仅是学好 C 语言程序设计的重要环节,而且是培养和考核计算机专业学生的动手能力、发现问题能力、分析问题和解决问题能力的重要环节。因此,C 语言程序设计的上机实验,是整个教学活动的重要环节。

本书的重要特点是适用面广:对于计算机专业本科学生,一般应该学完 1~10 章内容,如

果学有余力,还可以进一步学习和实践第 11 章的内容,从而进一步锻炼、提高自己的程序设计能力;对于计算机专业的专科学生和非计算机专业的学生,可以只学习、掌握 1~7 章的内容,而且只要真正掌握了这 7 章内容,就可以参加全国计算机等级考试的二级 C 语言考试。

本书的重要特点是概念清楚,内容丰富,每章配有练习题,便于教学和自学,特别是初学者,配合使用与之配套的教材,即《C 语言程序设计实验指导》,不仅可以按照实验内容和实验指导,自行上机编辑、调试和执行程序,而且还可以按照规范学会填写实验报告书,并提交给老师批改。对于具有一定基础的读者,也可以通过每个实验中的自编程练习题,独立编写程序,独立上机编辑、调试和执行程序,进一步提高自己的动手能力、发现问题的能力、分析问题和解决问题的能力。

本书由多所学校的教师合作编写,其中第 1 章由万臣编写,第 2 章由魏妮妮编写,第 3 章由贺杰编写,第 4 章由谢芳编写,第 5 章由刘智珺编写,第 6 章由彭媛编写,第 7 章和第 8 章由胡泉编写,第 9 章由王晶晶编写,第 10 章由董国刚编写,第 11 章由潘雪峰编写,附录由万臣编写。全书由胡金柱教授统稿。

由于各种原因,书中错误在所难免,敬请读者批评指正。

编者

2007 年 7 月

目 录

第 1 章 C 语言程序设计概论	(1)
1.1 C 语言的发展及其特点	(1)
1.1.1 C 语言的发展	(1)
1.1.2 C 语言的主要特点	(2)
1.2 C 语言程序结构	(2)
1.2.1 基本的 C 语言程序结构	(2)
1.2.2 C 语言书写规则	(4)
1.2.3 C 语言的字符集	(4)
1.2.4 C 语言词汇	(4)
1.3 C 程序的运行环境	(5)
1.3.1 上机操作的基本步骤	(5)
1.3.2 C 的运行环境	(6)
1.4 数据输出与输入	(8)
1.4.1 数据的输出	(8)
1.4.2 数据的输入	(10)
小结	(13)
习题	(13)
第 2 章 数据类型、运算符与表达式	(15)
2.1 C 语言的数据类型	(15)
2.2 常量、变量	(15)
2.2.1 常量	(15)
2.2.2 变量	(18)
2.3 C 语言的基本数据类型	(18)
2.3.1 整型数据	(18)
2.3.2 实型数据	(19)
2.3.3 字符型数据	(20)
2.3.4 数据类型转换	(21)
2.4 运算符与表达式	(22)
2.4.1 算术运算符与算术表达式	(22)
2.4.2 赋值运算符和赋值表达式	(23)
2.4.3 逗号运算符与逗号表达式	(24)
2.4.4 sizeof 运算符	(24)

2.4.5 运算符的优先级和结合性	(25)
2.5 简单程序设计与常见错误列举	(26)
2.5.1 简单程序设计	(26)
2.5.2 常见错误举例	(27)
小结	(28)
习题	(28)
第3章 选择结构程序设计	(30)
3.1 关系运算和逻辑运算	(30)
3.1.1 关系运算符和关系表达式	(30)
3.1.2 逻辑运算符和逻辑表达式	(31)
3.1.3 条件运算符和条件表达式	(32)
3.1.4 位运算符和位逻辑运算	(32)
3.2 程序的基本结构	(35)
3.2.1 程序中的数据结构	(35)
3.2.2 程序中的控制结构	(35)
3.3 if 语句	(37)
3.3.1 if 语句的三种形式	(37)
3.3.2 if 语句的嵌套	(42)
3.3.3 if 语句应用中应注意的问题	(44)
3.4 switch 语句	(45)
3.5 常见错误列举	(48)
小结	(50)
习题	(50)
第4章 循环结构程序设计	(53)
4.1 while 语句	(53)
4.2 do-while 语句	(55)
4.3 for 语句	(57)
4.4 循环嵌套	(60)
4.5 break 和 continue 语句	(63)
4.5.1 break 语句	(63)
4.5.2 continue 语句	(66)
4.6 goto 语句和标号	(67)
小结	(70)
习题	(71)
第5章 存储类型与数组	(74)
5.1 变量的存储类型与作用范围	(74)
5.1.1 变量的存储类型	(74)
5.1.2 局部变量与全局变量	(76)

5.2	一维数组	(78)
5.2.1	一维数组的定义与引用	(78)
5.2.2	一维数组的初始化	(79)
5.2.3	一维数组的程序举例	(80)
5.3	二维数组与多维数组	(84)
5.3.1	定义与初始化	(84)
5.3.2	二维数组的元素引用	(86)
5.3.3	多维数组	(90)
5.4	字符串与字符数组	(91)
5.4.1	字符数组	(91)
5.4.2	字符串	(93)
5.4.3	字符串数组	(94)
5.4.4	字符串处理函数	(96)
5.5	常见错误举例	(99)
小结		(100)
习题		(101)
第6章	函数与编译预处理	(103)
6.1	函数概述	(103)
6.1.1	C源程序的函数结构	(103)
6.1.2	C函数的分类	(104)
6.2	函数的定义与调用	(105)
6.2.1	函数的定义	(105)
6.2.2	函数的调用	(107)
6.2.3	函数的值	(110)
6.3	函数间数据的传递方式	(112)
6.3.1	传值调用	(112)
6.3.2	传址调用	(113)
6.3.3	数组作为函数参数	(114)
6.4	函数的嵌套调用和递归调用	(120)
6.4.1	函数的嵌套调用	(120)
6.4.2	递归函数与递归调用	(121)
6.5	编译预处理	(126)
6.5.1	宏定义	(127)
6.5.2	文件包含	(131)
6.5.3	条件编译	(132)
小结		(134)
习题		(134)
第7章	指针基础	(136)

7.1	指针的基本概念	(136)
7.2	指针变量	(137)
7.2.1	指针变量的定义	(137)
7.2.2	指针变量的引用	(137)
7.2.3	指针变量作为函数参数	(141)
7.2.4	指针变量的运算	(144)
7.3	指针与数组	(147)
7.3.1	指针与数组的联系	(147)
7.3.2	通过指针引用数组元素	(148)
7.3.3	指向数组的指针变量作函数参数	(150)
7.3.4	指向多维数组的指针变量	(157)
7.4	指针和字符串	(159)
7.4.1	字符串的表示	(159)
7.4.2	字符指针变量作函数参数	(163)
7.5	指针数组	(164)
7.5.1	指针数组的定义	(164)
7.5.2	指针数组的应用	(165)
小结		(168)
习题		(169)
第8章	指针与函数	(172)
8.1	指针参数	(172)
8.2	指针函数	(173)
8.2.1	指针函数的定义与调用	(174)
8.2.2	指针函数的应用	(175)
8.2.3	指针函数的数据传送与指针参数的区别	(176)
8.3	函数指针	(176)
8.3.1	函数指针变量的定义与赋值	(176)
8.3.2	函数指针的调用	(177)
8.4	多级指针	(180)
8.4.1	多级指针的定义	(180)
8.4.2	多级指针的应用	(181)
8.5	动态内存管理	(183)
8.5.1	动态内存管理概述	(183)
8.5.2	动态内存管理函数的应用	(185)
8.6	命令行参数	(186)
小结		(188)
习题		(188)
第9章	结构体与联合体	(192)

9.1 结构体	(192)
9.1.1 结构体的概念与定义	(192)
9.1.2 结构体类型变量的定义方法	(193)
9.1.3 结构体变量的引用	(195)
9.1.4 结构体变量的初始化	(196)
9.1.5 结构体数组	(197)
9.2 结构型指针与函数	(199)
9.2.1 结构指针	(199)
9.2.2 结构参数	(201)
9.2.3 结构型函数	(203)
9.2.4 结构指针型函数	(204)
9.3 线性链表	(205)
9.3.1 链表概述	(205)
9.3.2 动态链表	(206)
9.4 联合体	(211)
9.4.1 联合的定义	(211)
9.4.2 联合变量的定义	(211)
9.4.3 联合变量的引用	(212)
9.5 枚举类型变量的定义和引用	(214)
9.5.1 枚举类型的说明和枚举类型变量的定义	(214)
9.5.2 枚举类型变量的引用	(215)
9.6 typedef 的使用	(217)
小结	(218)
习题	(218)
第 10 章 文件	(221)
10.1 文件的基本概念与分类	(221)
10.1.1 文件的基本概念	(221)
10.1.2 文件的分类	(221)
10.2 文件结构体与文件指针	(223)
10.3 文件打开、读写与关闭	(224)
10.3.1 文件打开函数 fopen	(224)
10.3.2 文件关闭函数 fclose	(225)
10.3.3 文件读写函数	(226)
10.4 文件的定位	(233)
10.4.1 重返文件头函数	(233)
10.4.2 指针位置移动函数	(234)
10.4.3 取指针当前位置函数 ftell	(235)
10.5 非缓冲文件系统文件的基本操作	(235)

10.5.1 文件标识号与非缓冲文件系统	(235)
10.5.2 文件的创建函数、打开函数和关闭函数	(235)
10.5.3 文件读/写函数	(236)
10.5.4 文件定位函数	(237)
小结	(237)
习题	(238)
第 11 章 C 语言应用实例	(240)
11.1 简单应用实例	(240)
11.2 C 语言函数的应用实例	(250)
11.3 指针应用实例	(257)
11.4 使用颜色函数、图形函数和时间函数实例	(264)
11.5 文件使用实例	(267)
附录一 ASCII 代码表	(277)
附录二 常用 Turbo C 2.0 标准函数	(279)
附录三 ACM 大赛简介	(283)
附录四 程序员考试简介	(284)
附录五 计算机等级考试简介	(285)
参考文献	(286)

第 1 章 C 语言程序设计概论

C 语言是国际上广泛流行、很受欢迎的高级程序设计语言之一,它既可以用来编写系统软件,又可以用来编写应用软件。本章主要介绍 C 语言的发展、C 语言的特点、C 语言的结构、C 语言的上机步骤与环境以及 C 语言的数据输入和输出。通过本章的学习,读者应重点掌握 C 语言的特点、C 语言的结构、上机运行 C 程序的方法以及 C 语言数据的输入和输出方法。

1.1 C 语言的发展及其特点

C 语言是目前世界上最流行、使用最广泛的计算机高级语言之一。在设计操作系统等系统软件和需要对硬件进行操作时,使用 C 语言明显优于其他计算机语言,许多大型应用软件和系统软件都是用 C 语言编写的。

1.1.1 C 语言的发展

C 语言的起源可以追溯到 1960 年出现的 ALGOL 60。1963 年英国的剑桥大学在 ALGOL 60 语言的基础上推出了 CPL(Combined Programming Language)语言,但是 CPL 语言难以实现。1967 年,剑桥大学的 Martin Richards 对 CPL 语言进行了简化,推出了 BCPL 语言。1970 年,美国贝尔实验室的 Ken Thompson 以 BCPL 语言为基础,又做了进一步的简化,设计出了很简单且接近硬件的 B 语言,并用 B 语言写了第一个 UNIX 操作系统,在 DEC PDP-7 型计算机上实现。1971 年在 DEC PDP-11 型计算机上实现了 B 语言。1972 年由美国的 Dennis M. Ritchie 在 B 语言的基础上设计出了 C 语言,并首次在配置 UNIX 操作系统的 DEC PDP-11 型计算机上使用。

为了推广 UNIX 操作系统,1977 年 Dennis M. Ritchie 发表了不依赖于具体机器系统的可移植的 C 语言编译程序。1978 年 Brian W. Kernighan 和 Dennis M. Ritchie 出版了《The C Programming Language》一书,从而使 C 语言成为目前世界上流行最广泛的高级程序设计语言之一。

随着微型计算机的日益普及,出现了许多 C 语言版本。由于没有统一的标准,使得这些 C 语言之间出现了一些不一致的地方。为了改变这种状况,1983 年美国国家标准化协会(ANSI)为 C 语言制定了第一个 ANSI 标准,称为 ANSI C。1987 年 ANSI 又公布了新的 C 语言标准,称为 87 ANSI C。这个标准于 1989 年被国际标准化组织(ISO)采用,被称为 ANSI/ISO Standard C(即 C89)。目前国际上流行的 C 编译系统都是以这个标准为基础的。

1995 年 ANSI 又为 C 语言增加了一些新的函数,使之具有 C++ 的一些特征,使 C89 成为 C++ 的子集。1999 年推出的 C99 在保留 C 语言特征的基础上,增加了一系列面向对象的新特征,C 语言也从面向过程的语言发展成为面向对象的语言。

C 语言已经成为全球程序员所使用的公共编程语言,并由此产生了当前程序设计的两个主流语言 C++ 和 Java。它们都是建立在 C 语言的语法和基本结构基础上的,而且现在世界上的许多软件都是在 C 语言及其衍生的各种语言的基础上开发而形成的。所以 C 语言是学

习 C++ 和 Java 语言的基础语言,每一个大学生都应该努力学会用 C 语言编写程序。

1.1.2 C 语言的主要特点

一种语言之所以能存在和发展,并具有生命力,总是有其不同于(或者优于)其他语言的特点。C 语言的主要特点可归纳为以下 7 点:

(1) 语言简洁、紧凑,使用方便、灵活。C 语言一共只有 32 个关键字,9 种控制语句,程序书写形式自由,主要用小写字母表示,压缩了一切不必要的成分。

(2) 运算符丰富。C 语言的运算符涉及的范围十分广泛,共有 34 种运算符。C 语言把括号、赋值、强制类型转换等都作为运算符来处理,从而使 C 语言的运算类型极其丰富,表达式类型也多样化。灵活地使用各种运算符可以实现在其他高级语言中难以实现的运算。

(3) 数据结构丰富,具有现代化语言的各种数据结构。C 语言的数据类型有整型、实型、字符型、数组类型、指针类型、结构体类型、共用体类型等。能用来实现各种复杂的数据结构(如链表、树、堆栈等)的运算,尤其是指针类型数据,使用起来比其他语言(如 PASCAL)更为灵活、多样。

(4) C 语言是一种结构化的程序设计语言,它层次清晰,便于按模块化方式来组织程序,易于调试和维护。例如,它有 if-else 语句、while 语句、do-while 语句、switch 语句、for 语句等结构化的控制语句。用函数作为程序的模块单位,便于实现程序的模块化。

(5) C 语言允许直接访问物理地址,能进行位(bit)操作,能实现汇编语言的大部分功能,可直接对硬件进行操作。因此,C 语言既具有高级语言的功能,又具有低级语言的许多功能,可用来编写系统软件。C 语言的这种双重性质,使它既是成功的系统描述语言,又是通用的程序设计语言。有人把 C 语言称为“高级语言中的低级语言”,或“中级语言”,其原因是它兼有高级语言和低级语言的特点。

(6) 生成的目标代码质量高,程序执行效率高。一般只比汇编程序生成的目标代码效率低 10%~20%。

(7) 用 C 语言写的程序可移植性好(与汇编语言相比)。基本上不做修改就能用于各种型号的计算机和各种操作系统。

C 语言具有以上的许多优点,但不是完美无缺的,它的缺点表现在:

变量类型的转换比较随便,不够安全,例如 C 语言程序中字符型变量数据与整型变量数据可以通用;另外,C 语言编译系统对程序的语法检查不太严格,例如对数组的下标越界不做检查;还有运算符的优先级太多,容易混淆,程序的正确性要由程序的编制者自己来判断。

目前,在微型计算机上广泛使用的 C 语言编译系统有 Turbo C、Borland C++、Microsoft Visual C++ 等。虽然它们的基本部分都是相同的,但还是有一些细微的差异,本书采用 Turbo C 作为上机编程调试环境。

1.2 C 语言程序结构

1.2.1 基本的 C 语言程序结构

【例 1.1】 用 C 语言实现输出字符串“Welcome to C world!”。

```
#include <stdio.h>
void main( )                /* 主函数 */
{
```

```
printf("Welcome to C world! \n");    /* 调用标准函数,显示引号中的内容 */
}
```

这是一个最简单的 C 程序,其执行结果是在屏幕上显示一行信息:

Welcome to C world!

程序中第一行“# include <stdio. h>”,是将头文件 stdio. h 嵌入到本程序中,以便程序在执行过程中调用输入、输出函数。C 语言没有输入输出语句,输入输出通过调用系统提供的函数来完成。上述程序中,第二行到第五行是一个主函数(main 函数),介于第三行的左花括号“{”和第五行的右花括号“}”之间的部分称为函数体。“{”表示函数体开始,“}”表示函数体结束。其中第四行是一个输出函数(printf 函数),它的功能是把要输出的内容送到显示器去显示。printf 函数是系统定义的标准函数,可在程序中直接调用。双引号引住的内容就是输出的字符串。“\n”表示换行。

【例 1.2】 求两个数的和。

```
#include <stdio. h>

void show(int x,int y);    /* (自定义)函数原型声明 */
void main( )               /* main 函数定义 */
{ int a,b;
  a=1; b=2;
  show(a,b);
}

void show(x,y)              /* (自定义)函数定义 */
int x,y;
{ int sum;
  sum=x+y;
  printf(" a add b is %d \n",sum);
}
```

本程序的功能是计算两个数 a、b 的和,并输出。本程序由两个函数组成,即主函数和 show 函数,两个函数之间是并列关系,从主函数中调用 show 函数。show 函数的主要功能是计算并显示主函数中 a、b 的值的和。show 函数是一个用户自定义函数,因此在主函数调用它之前要进行函数原型声明。具体的函数内容将在第 6 章详细叙述。

上述程序的输出结果是:

a add b is 3

由上面两个例子的程序可以明显地看出 C 语言程序的结构特点如下:

- (1) 一个 C 语言源程序可以由一个或多个源文件组成。
- (2) 每个源文件可由一个或多个函数组成。
- (3) 一个源程序不论由多少个文件组成,都有一个且只能有一个 main 函数,即主函数。任何一个完整的 C 程序都必须有这个主函数,不管这个主函数出现在程序中的什么位置,C 程序的执行都从主函数开始。

(4) 源程序中可以有预处理命令(include 命令仅为其中的一种),预处理命令通常应该放在源文件或源程序的最前面。

(5) 每一个语句都必须以分号(;)结尾。但预处理命令、函数头和花括号“{”之后不能加

分号。

(6) 标识符、关键字之间必须至少加一个空格以示间隔。若已有明显的间隔符,也可不再加空格来间隔。

(7) 可以用 `/* ..... */` 做注释。其格式必须是: `/* 注释内容 */`。应当注意的是,在 `/` 和 `*` 之间不允许插入空格或其他字符。注释的内容可以用英文、汉字或汉语拼音书写。注释语句的位置可以放在程序的开头,放在某一行的后面或者在任何两行之间单独占用一行也都是允许的。注释的内容通常是关于程序的功能、算法或者某些注意事项等方面的文字说明。程序中插入注释的目的是帮助阅读和理解程序,增强程序的可读性。注释在编译时不产生目标代码,仅作为源程序的一部分驻留在源程序清单中。

1.2.2 C 语言书写规则

1. 一般情况下,一个说明或一个语句占一行。

2. 用 `{ }` 括起来的部分,通常表示程序的某一层结构。`{ }` 一般与该结构语句的第一个字母对齐,并单独占一行。

3. 低一层次的语句可比高一层次的语句或说明缩进若干空格后书写,以便使程序结构看起来更加清晰,增加程序的可读性。

在编写程序时应力求遵循这些规则,以养成良好的编程习惯。

1.2.3 C 语言的字符集

字符是组成语言的最基本的元素。C 语言字符集由字母、数字、空格、标点和特殊字符组成。在字符常量、字符串常量和注释中还可以使用汉字或其他可表示的图形符号。

1. 字母:小写字母 `a~z` 共 26 个,大写字母 `A~Z` 共 26 个。

2. 数字:`0~9` 共 10 个。

3. 空白符:空格符、制表符、换行符等统称为空白符。空白符只在字符常量和字符串常量中起作用。在其他地方出现时,只起间隔作用,编译程序对它们忽略。因此在程序中使用空白符与否,对程序的编译不发生影响,但在程序中适当的地方使用空白符可以增加程序的清晰性和可读性。

4. 标点符号:`' " ; :`

5. 特殊符号:`\ _ $ #`

1.2.4 C 语言词汇

在 C 语言中使用的词汇分为六类:标识符、关键字、运算符、分隔符、常量、注释符等。

1. 标识符

在程序中使用的变量名、函数名统称为标识符。除库函数的函数名由系统定义外,其余都由用户自定义。C 语言规定,标识符只能是字母(`A~Z, a~z`)、数字(`0~9`)、下划线(`_`)组成的字符串,并且其第一个字符必须是字母或下划线。

以下标识符是合法的:

`x`

`_3x`

`BOOK_1`

`sum5`

以下标识符是非法的:

`3s` 以数字开头

`s * T` 出现非法字符 `*`

`-3x` 以减号开头

`bowy-1` 出现非法字符 `-`(减号)

在使用标识符时必须注意以下几点:

(1) 标准 C 不限制标识符的长度,但它受各种版本的 C 语言编译系统限制,同时也受到具

体机器的限制。例如在某版本C中规定标识符前八位有效,当两个标识符前八位相同时,则被认为是同一个标识符。例如:sumx123456和sumx12349876是同一个标识符。

(2) 在标识符中,大、小写字母是有区别的。例如:BOOK和book是两个不同的标识符。

(3) 标识符虽然可由程序员随意定义,但标识符是用于标识某个量的符号。因此,命名应尽量具有相应的意义,以便阅读理解,做到“顾名思义”。

2. 关键字

关键字是由C语言规定的具有特定意义的字符串,通常也称为保留字。用户定义的标识符不应与关键字相同。C语言的关键字分为以下几类:

(1) 类型说明符

用于定义、说明变量、函数或其他数据结构的类型。如int,double等。

(2) 语句定义符

用于表示一个语句的功能。如if-else就是条件语句的语句定义符。

(3) 预处理命令字

用于表示一个预处理命令。如前面各例中用到的include。

3. 运算符

C语言中含有相当丰富的运算符。运算符与变量、函数一起组成表达式,表示各种运算功能。运算符由一个或多个字符组成。具体的运算符将在第2章中详细介绍。

4. 分隔符

在C语言中采用的分隔符有逗号和空格两种。逗号主要用在类型说明和函数参数表中分隔各个变量。空格多用于语句各单词之间做间隔符。在关键字和标识符之间必须要有一个以上的空格符做间隔,否则将会出现语法错误。例如把“int a;”写成“inta;”,C编译器会把inta当成一个标识符处理,其结果必然出错。

5. 常量

C语言中使用的常量可分为数字常量、字符常量、字符串常量、符号常量、转义字符等多种,在第2章中将专门给予介绍。

6. 注释符

C语言的注释符是以“/*”开头并以“*/”结尾的串。在“/*”和“*/”之间的即为注释的内容。程序编译时,不对注释作任何处理。注释可出现在程序中的任何位置。注释用来向用户提示或解释程序的意义。

1.3 C程序的运行环境

1.3.1 上机操作的基本步骤

在计算机上开发、运行一个C程序,一般都包括以下四步:

1. 程序的编辑

在上机操作之前先编写好解决某个问题的C程序,然后用一种编辑软件(编辑器)将编写好的C程序输入计算机,并以文本文件的形式保存到计算机的磁盘上。编辑的结果是建立C源程序文件,该文件的后缀名必须为“.c”(一般由编辑器自动加上)。

C程序习惯上使用小写英文字母,常量和用途的符号可用大写字母。C语言中大、小写字母是有区别的,关键字必须小写。

2. 程序的编译

编译是指将编辑好的源文件翻译成二进制目标代码的过程。编译过程是使用 C 语言提供的编译程序(编译器)完成的。不同操作系统下的各种编译器的使用命令不完全相同,使用时应注意计算机环境。编译时,编译器首先要对源程序中的每一个语句检查语法错误,当发现错误时,会在屏幕上显示错误的位置和错误类型的信息。此时,要再次调用编辑器进行查错修改。然后,再进行编译,直至排除所有语法和语义错误。正确的源程序文件经过编译后会在磁盘上自动生成目标文件(后缀名为 .obj 的文件)。

3. 程序的连接

编译后产生的目标文件是可重定位的程序模块,不能直接运行。连接就是把目标文件和其他必需的目标程序模块(这些目标程序模块是分别编译而生成的目标文件)以及系统提供的标准库函数链接在一起,生成可以运行的可执行文件的过程。连接过程使用 C 语言提供的连接程序(连接器)来完成,生成的可执行文件(后缀名为 .exe 的文件)将存放到磁盘中。

4. 程序的运行

程序运行生成可执行文件后,就可以在操作系统控制下运行。若执行程序后达到预期目的,则 C 程序的开发工作到此完成。否则,要进一步检查、修改源程序,重新进行编辑—编译—连接—运行,直到取得预期结果为止。

上述介绍的一个完整的 C 程序的上机操作步骤如图 1-1 所示。

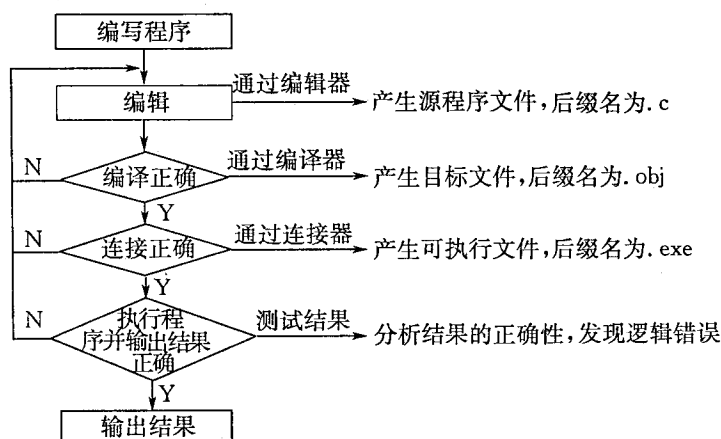


图 1-1 C 程序上机操作步骤流程示意图

1.3.2 C 的运行环境

Turbo C 是美国 Borland 公司推出的 IBM PC 系列机的 C 语言编译程序。它具有方便、直观、易用的界面和丰富的库函数。并向用户提供了集成环境,把程序的编辑、编译、连接和运行等操作都集中在一个界面上进行,方便了用户的使用。

1. Turbo C 工作环境简介

一个 C 程序的编辑、调试是从进入 Turbo C 的集成环境开始的。而进入 C 语言的环境,一般有两种途径:从 DOS 环境进入和从 Windows 环境进入(假设 Turbo C 的安装目录为 C:\TC)。

(1) 从 DOS 环境进入

在 DOS 命令行上键入:

C:\>CD TC (指定当前目录为 TC 子目录,其中“ ”表示空格,“ ”表示回车,全