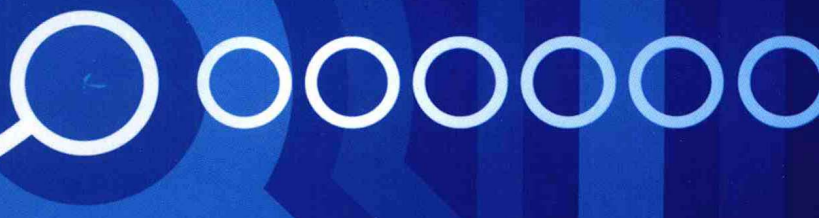


高等院校通用教材



语言程序设计

主 编 邓春伟



吉林教育出版社

C 语言程序设计

主 编：邓春伟

副主编：(姓名不分先后)

王 聘 王树芬 王恭礼

主 审：王克家

编 委：(姓名不分先后)

王志伟 韩大伟 王永辉

刘 强

图书在版编目(CIP)数据

C 语言程序设计/邓春伟著. —长春:吉林教育出版社,

2008.12

ISBN 978—7—5383—5564—2

I. C… II. 邓… III. C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(2008)第 183022 号

责任编辑 孙华群

装帧设计 王洪义

-
- ◇出版 吉林教育出版社 电话 0431—85645386
◇发行 吉林教育出版社 电话 0431—85645386
◇社址 长春市同志街 1991 号 邮政编码 130021
◇印刷 长春市博文印刷厂
◇开本 787 毫米×960 毫米 16 开
◇印张 14.25
◇字数 245 千字
◇版次 2008 年 12 月第 1 版 2008 年 12 月第 1 次印刷
◇书号 ISBN 978—7—5383—5564—2
-

定 价:35.00 元

版权所有 侵权必究 印装差错 负责调换 打击盗版 举报有奖

前 言

C 语言是一种应用十分广泛的计算机语言，其功能丰富、表达能力强、使用灵活方便、应用面广、目标程序效率高、可移植性好，既具有高级语言的优势，又具有低级语言的许多特点，特别适合编写系统软件，已经成为计算机专业以及非计算机专业工科学生、高职高专学生及中专学生的必修课程。

本书符合教学大纲的基本要求，在编写过程中参考了《全国计算机等级考试二级考试大纲》中有关 C 语言程序设计的要求。在内容上突出以实用为导向，以实践技能为核心，注重全面提高学生的职业实践能力和素养。在内容上力求准确精要、层次清晰、通俗易懂、实用性强，使学生在少走弯路的前提下对 C 语言产生浓厚的学习兴趣。

全书共分 10 章，主要介绍了 C 语言程序设计基础知识、运算符和表达式、三种基本结构的程序设计方法、数组、函数、指针、结构体与共用体和文件等。在每章之后提供的习题和实训内容，突出了实用性，强调理论与实践相结合，培养了学生解决实际问题的能力。

本书不仅可以作为独立院校计算机及相关专业的教材，而且可以作为“全国计算机等级考试二级 C 语言程序设计”的辅导教材，或作为自学 C 语言的参考用书。

本书由邓春伟主编，王克家主审，全书各章节的编写分工如下：第 1 章、第 8 章由王聘编写；第 2 章、第 7 章、第 9 章由邓春伟编写；第 3 章、第 5 章由王树芬编写；第 4 章、第 6 章由王恭礼编写；第 10 章由王志伟、韩大伟、王永辉、刘强编写。

由于编者水平有限，错误在所难免，请广大读者批评指正。

目 录

第1章 C语言概述	(1)
1.1 C语言历史	(1)
1.2 C语言的特点	(2)
1.3 简单的C语言源程序介绍	(3)
1.4 程序设计的灵魂——算法	(6)
1.4.1 什么是算法	(6)
1.4.2 算法的特征	(7)
1.4.3 算法的表示	(7)
1.5 C程序的上机步骤	(10)
本章小结	(11)
习 题	(11)
第2章 C语言基础知识	(14)
2.1 关于C语言的数据类型	(14)
2.2 标识符号	(15)
2.3 数据	(16)
2.3.1 常量	(16)
2.3.2 变量	(19)
2.3.3 变量赋初值	(21)
2.3.4 各种类型混合运算	(21)
2.4 常用的输入\输出语句	(23)
2.4.1 字符输入\输出函数	(23)
2.4.2 格式输入\输出函数	(24)
本章小结	(28)

习 题	(28)
第3章 运算符及其表达式	(35)
3.1 算术运算符及其算术表达式	(36)
3.1.1 表达式	(36)
3.1.2 算术运算符及其表达式	(36)
3.2 自增、自减运算符	(38)
3.2.1 自增、自减运算符 ++、--	(38)
3.2.2 有关自增、自减运算符的说明	(40)
3.3 赋值运算符及其赋值表达式	(40)
3.3.1 赋值运算符	(40)
3.3.2 常用形式	(41)
3.3.3 赋值表达式及赋值运算符的优先级与结合性	(41)
3.3.4 赋值过程中的类型转换	(42)
3.4 关系运算符及其表达式	(44)
3.4.1 关系运算符及其优先级	(44)
3.4.2 关系表达式	(45)
3.5 逻辑运算符及逻辑表达式	(45)
3.5.1 逻辑运算符及其优先关系	(45)
3.5.2 逻辑表达式	(47)
3.6 位运算符	(49)
3.7 其他运算符	(51)
本章小结	(53)
习 题	(53)
第4章 C 程序设计——顺序和选择结构程序设计	(55)
4.1 C 结构化程序设计	(55)
4.1.1 结构化程序设计思想	(55)
4.1.2 结构化程序设计的三种基本结构	(56)
4.2 C 语言中的语句	(58)
4.3 顺序结构程序设计	(60)
4.4 选择结构程序设计	(61)

4.4.1	if 语句	(62)
4.4.2	switch 语句	(69)
4.5	顺序结构和选择结构程序设计举例	(72)
4.5.1	顺序结构程序设计举例	(72)
4.5.2	选择结构程序设计举例	(74)
本章小结		(77)
习 题		(77)
第 5 章	循环结构程序设计	(79)
5.1	循环结构控制语句	(79)
5.2	循环嵌套	(86)
5.3	break 和 continue 语句	(88)
5.4	程序举例	(91)
本章小结		(95)
习 题		(96)
第 6 章	数 组	(99)
6.1	一维数组的定义和引用	(99)
6.1.1	一维数组的定义方式	(99)
6.1.2	一维数组元素的引用	(101)
6.1.3	一维数组的初始化	(102)
6.1.4	一维数组程序举例	(103)
6.2	用数组实现排序问题	(104)
6.2.1	冒泡(起泡)排序法	(104)
6.2.2	选择排序法	(106)
6.2.3	插入排序法	(108)
6.3	二维数组的定义和引用	(109)
6.3.1	二维数组的定义	(109)
6.3.2	二维数组元素的引用	(110)
6.3.3	二维数组的初始化	(111)
6.3.4	二维数组程序举例	(112)
6.4	字符数组	(114)

6.4.1 字符数组的定义	(114)
6.4.2 字符数组的初始化	(115)
6.4.3 字符数组的引用	(115)
6.4.4 字符串和字符串结束标志	(116)
6.4.5 字符数组的输入输出	(116)
6.4.6 字符串处理函数	(118)
6.4.7 字符数组程序举例	(122)
本章小结	(123)
习 题	(123)
第7章 函 数	(125)
7.1 引 言	(125)
7.2 函数的定义	(125)
7.2.1 函数的分类	(126)
7.2.2 函数的定义	(126)
7.3 函数参数及其函数的值	(127)
7.4 函数的调用	(129)
7.4.1 函数的调用	(129)
7.4.2 函数的声明	(129)
7.4.3 函数的嵌套调用	(130)
7.5 函数递归调用	(131)
7.6 数组作为函数参数	(133)
7.6.1 数组元素作函数实参	(133)
7.6.2 数组名作函数参数	(133)
7.7 程序举例	(135)
7.8 变量的作用域	(138)
7.8.1 局部变量	(138)
7.8.2 全局变量	(139)
7.9 变量的存储类别	(141)
本章小结	(143)
习 题	(143)

第8章 指 针	(145)
8.1 指针的基本概念	(145)
8.1.1 指针的概念	(145)
8.1.2 指针变量的定义与初始化	(147)
8.1.3 指针的运算及引用	(148)
8.1.4 指针作为函数参数	(151)
8.2 指针与数组	(154)
8.2.1 指针与一维数组	(154)
8.2.2 指针与字符串	(158)
8.3 指针与函数	(163)
8.3.1 指向函数的指针	(164)
8.3.2 返回指针的函数	(165)
8.3.3 带参数的主函数	(167)
8.4 程序举例	(168)
本章小结	(173)
习 题	(176)
第9章 结构体与共用体	(177)
9.1 结构体变量的定义及引用	(177)
9.1.1 结构体变量的定义	(177)
9.1.2 结构体变量的引用	(179)
9.1.3 结构体变量的初始化及结构体数组	(180)
9.2 指向结构体类型数据的指针	(181)
9.2.1 结构体类型的指针变量	(181)
9.2.2 用指向结构体的指针作函数参数	(182)
9.3 共用体	(185)
9.3.1 共用体的定义	(185)
9.3.2 共用体变量的引用及特点	(186)
9.3.3 共用体变量的应用	(187)
9.4 枚举类型	(188)
9.4.1 枚举类型的定义和枚举变量的说明	(188)

9.4.2 枚举类型变量的赋值和使用	(189)
本章小结	(191)
习 题	(192)
第 10 章 文 件	(193)
10.1 C 文件概述	(193)
10.2 文件指针	(195)
10.3 文件的打开与关闭	(196)
10.3.1 文件的打开(fopen 函数)	(196)
10.3.2 文件的关闭(fclose 函数)	(197)
10.4 文件的读写	(198)
10.4.1 字符读写函数 fgetc 和 fputc	(198)
10.4.2 数据块读写函数 fread 和 fwrite	(199)
10.5 文件的随机读写	(201)
10.5.1 文件定位	(201)
10.5.2 文件的随机读写	(202)
10.6 文件检测函数	(203)
本章小结	(204)
习 题	(204)
附录 1 C 语言关键字	(205)
附录 2 运算符和结合性	(206)
附录 3 常用字符与 ASCII 代码对照表	(208)
附录 4 C 库函数	(211)

第1章 C语言概述

本章主要介绍 C 语言的发展历史、基本特点、简单的 C 语言程序结构和算法以及 C 语言的上机步骤。其中 C 语言程序结构和算法是重点，是以后各章学习的基础。

1.1 C语言历史

C 语言是国际上广泛流行的、很有发展前途的计算机高级语言。它适合作为系统描述语言，既可用于编写系统软件，也可用来编写应用软件。

C 语言的发展颇为有趣。它的原型 ALGOL 60 语言(也称为 A 语言)。1963 年，剑桥大学将 ALGOL 60 语言发展成为 CPL(Combined Programming Language)语言。此语言在 ALGOL 语言的基础上增加了硬件处理能力。同年，剑桥大学的 Martin Richards 对 CPL 语言进行了简化，于是产生了 BCPL 语言。1970 年，美国贝尔实验室的 Ken Thompson 将 BCPL 进行了修改，并为它起了一个有趣的名字“B 语言”。意思是将 CPL 语言煮干，提炼出它的精华。并且他用 B 语言写了第一个 UNIX 操作系统。而在 1973 年，B 语言也给人“煮”了一下，美国贝尔实验室的 D. M. Ritchie 在 B 语言的基础上最终设计出了一种新的语言，他取了 BCPL 的第二个字母作为这种语言的名字，这就是 C 语言。

为了使 UNIX 操作系统推广，1977 年 Dennis M. Ritchie 发表了不依赖于具体机器系统的 C 语言编译文本《可移植的 C 语言编译程序》。1978 年 Brian W. Kernighian 和 Dennis M. Ritchie 出版了名著《The C Programming Language》，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。1988 年，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。为了改变这种情况，美国国

家标准研究所(ANSI)为 C 语言制定了一套 ANSI 标准,成为现行的 C 语言标准。

自 1972 年投入使用之后, C 语言成为 UNIX 和 XENIX 操作系统的主要语言。是当今最为广泛使用的程序设计语言之一。

本书以 Turbo C 2.0 为学习的平台,各章例题均在 Turbo C 2.0 环境下调试运行。

1.2 C 语言的特点

C 语言具有以下基本特点:

1. C 语言是具有低级语言功能的高级语言

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。它把高级语言的基本结构和语句与低级语言的实用性结合起来,是处于汇编语言和高级语言之间的一种程序设计语言,也可称其为“中级语言”。

2. C 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, Turbo C 2.0 增加了 11 个关键词(用于各种增强和扩展功能)。C 语言可以像汇编语言一样对位、字节和地址进行操作,而这三者是计算机最基本的工作单元。程序书写形式自由,主要用小写字母表示,相对于其他高级语言源程序较短。C 语言语法限制不太严格、程序设计自由度大。

3. 运算符丰富,表达式能力强

C 语言共有 34 种运算符,范围广泛,除一般高级语言所使用的算术、关系和逻辑运算符外,还可以实现以二进制位为单位的运算,并且具有如 `a++`, `--b` 等单项运算符和 `+=`、`-=`、`*=`、`/=` 等复合运算符等。

4. 数据结构丰富

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类型的运算,并引入了指针概念,使程序效率更高。另外 C 语言具有强大的图形功能,支持多种显示器和驱动器,且计算功能、逻辑判断功能强大。

5. C 语言是结构式语言

结构式语言的显著特点是代码及数据的分隔化,即程序的各个部分除了必要的信息交流外彼此独立。这种结构化方式可使程序层次清晰,便于使用、

维护以及调试。C语言是以函数形式提供给用户的，这些函数可方便地调用，并具有多种循环、条件语句控制程序流向，从而使程序完全结构化。

6. C语言允许直接访问物理地址，可以直接对硬件进行操作

因此既具有高级语言的功能，又具有低级语言的许多功能，能够像汇编语言一样对位、字节和地址进行操作，而这三者是计算机最基本的工作单元，可以用来写系统软件。

7. C语言适用范围大，可移植性好

C语言有一个突出的优点就是适合于多种操作系统，如DOS、UNIX，也适用于多种机型。

8. C语言也存在一些不足之处，例如运算符及其优先级过多、语法定义不严格等，对于初学者有一定的困难。

C语言具有上述特点，因此C语言得到了迅速推广，成为人们编写大型软件的首选语言之一。许多原来用汇编语言处理的问题可以用C语言来处理了。

1.3 简单的C语言源程序介绍

一个完整的C语言程序由一个或多个具有相对独立功能的程序模块组成，这样的程序模块称为“函数”。因此，函数是C程序的基本单位。

一个C程序，不管它有多简单，都必须有且只有一个主函数，例如：

```
main()
```

```
{  
}
```

上面的函数是一个空主函数，程序没有任何意义，它什么都不做。

我们先通过一个简单的C程序，来了解C程序。

【例1.1】在计算机屏幕上输出“Hello, World!”。

```
#include <stdio.h>
```

```
main()
```

```
{  
    printf("Hello, World!\n");  
}
```

运行结果为：Hello, World!

“`#include <stdio.h>`”称为命令行，命令行必须用“`#`”开头，后面不能加“`;`”，因为它不是 C 程序中的语句，`stdio.h` 是系统提供的头文件，其中包含有关输入输出函数的信息。这个源程序中 `main` 是主函数名，C 语言规定必须用 `main` 作为主函数名，函数名后的一对圆括号不能省略，圆括号中内容可以是空的。一个 C 程序可以包含任意多个函数，但必须有且只有一个主函数。一个 C 程序总是从主函数开始执行，最后在主函数结束。函数体需用花括号括起来，左括号表示函数体的开始，右括号表示函数体的结束。其间可以有定义（说明）部分和执行语句部分；每一条语句都必须用分号“`;`”结束，语句的数量不限，程序中由这些语句向计算机系统发出指令，本程序函数体内只有一条输出语句，双引号内的内容原样输出，“`\n`”表示输出字符后换行。

【例 1.2】已知圆的半径，求圆的周长和面积。

```
#include <stdio.h>

main()
{
    int r;                /* 说明圆半径 r 为整型变量 */
    float l,s;            /* 说明周长 l 面积 s 为实型变量 */
    r=5;                  /* 给半径 r 赋初值 */
    l=2*3.14159*r;        /* 计算 l 的值 */
    s=3.14159*r*r;        /* 计算 s 的值 */
    printf("r=%d, l=%f, s=%f\n",r,l,s); /* 输出圆的半径、周长和面积 */
}
```

运行结果为：`r=5, l=31.415899, s=78.539749`

程序中首先定义了 3 个变量，其中 `r` 为整型变量，`l`、`s` 为实型变量。然后设置 `r` 的值，并根据 `r` 的值计算圆周长和面积。输出语句中的“`%d`”，“`%f`”为输出格式符，分别表示十进制整型和实型，它指定输出结果时的数据类型和格式，程序在执行时，该位置由具体数据替代。

程序中的“`/* …… */`”表示注释部分，作用是帮助用户阅读程序，它对程序的运行不起作用，在对源程序进行编译时，注释会被忽略。“`/*`”和“`*/`”必须成对出现，且“`/`”和“`*`”之间不能有空格，注释内容可以是西文，也可以是中文，注释通常用于说明变量的含义、程序段的功能。注释部分可以放在程序中任意合适位置，一个好的程序应该有必要的注释，这样可以增加可读性。

【例 1.3】输入矩形的两个边长，求矩形的面积。

```
#include "stdio. h"

main()
{
    int  x, y, z;
    scanf( "%d,%d", &x, &y);      /* 输入矩形的两条边长 */
    z = area(x, y);                /* 调用函数 area */
    printf( "area  is %d \n", z);   /* 输出矩形的面积 */
}

int  area(int a, int b)            /* 定义子函数求矩形的面积 */
{int c;
  c = a * b;
  return (c);                     /* 返回矩形面积的值 */
}
```

运行结果: 4, 5 ↙

area is 20

本程序由主函数 main 和被调用函数 area 组成，在主函数中输入两条边长 x、y，然后通过语句 `z = area(x, y)` 调用函数 area，计算结果由 return 语句返回给主函数。这两个函数在位置上是独立的，可以把主函数 main 放在前面，也可以把主函数 main 放在函数 area 的后面。

scanf 和 printf 是 C 语言提供的标准输入输出函数，&a 和 &b 中的“&”的含义是“取地址”，程序中 scanf 函数的作用是将从键盘上键入的两个数，输入到变量 x 和 y 所标志的内存单元中，亦即输入给变量 x 和 y。

通过以上三个 C 程序的例子，可以看出 C 程序的特点如下：

1. C 程序由函数组成

每个 C 程序有且仅有一个主函数，该主函数的函数名规定为 main，也可以包含一个 main 函数和若干个子函数。

2. 每个函数的定义分为两部分：函数说明和函数体

函数说明的形式：

函数类型 函数名(形式参数 1 类型 形式参数 1, 形式参数 2 类型 形式参数 2, ……)

函数体的形式：

```
{
```

变量定义(说明)部分

函数执行部分

}

3. C 程序的书写格式自由,一行内可以写几条语句,一条语句也可以写在多行上,每条语句后必须以“;”作为语句的结束。复合语句要以一对{}括起来。

4. C 程序的执行总是从主函数开始,并在主函数中结束。主函数的位置在程序中是任意的,其他函数总是通过函数调用语句来执行。

5. 主函数可以调用任何非主函数,任何非主函数都可以相互调用,但是不能调用主函数。

6. C 语言本身没有输入输出语句。输入和输出操作是由调用系统提供的输入输出函数来完成的。

7. 可以用/*……*/对 C 程序中的任何部分作注释。

8. C 程序严格区分大小写,关键字必须小写。

9. 用一对大括号{}来标识一个函数或复合语句的范围。

1.4 程序设计的灵魂——算法

1.4.1 什么是算法

算法是指解决问题的方法和步骤。在计算机中,算法是一系列解决问题的清晰指令,也就是说,能够对一定规范的输入,在有限时间内获得所要求的输出。

利用计算机解决问题,首先要编写计算机程序。计算机程序是许多指令的集合,每一条指令让计算机执行完成一个具体的操作,一个程序所规定的操作全部执行完后,就能产生计算结果。因此,编写出正确的程序是让计算机解决实际问题的关键。一般编制正确的计算机程序必须具备两个基本条件:一是掌握一门计算机高级语言的规则,二是要掌握解题的方法和步骤。

计算机语言只是一种工具。单单掌握语言的语法规则是不够的,最重要的是学会针对各种类型的问题,拟定出有效的解题方法和步骤的算法。

【例 1.4】算法描述 $1 + 2 + 3 + \cdots + 100$ 之和。

S1: 使 $\text{sum} = 0$;

S2: 使 $i = 1$;
 S3: 使 $\text{sum} + i$, 和仍然放在 sum 中, 可表示为 $\text{sum} + i \Rightarrow \text{sum}$;
 S4: 使 i 的值加 1, 可表示为 $i + 1 \Rightarrow i$;
 S5: 如果 i 不大于 100; 返回步骤 3 及其后的步骤 4 和 5; 否则, 算法结束。最后 sum 里的值就是 $1 + 2 + 3 + \cdots + 100$ 之和。

1.4.2 算法的特征

一个正确的算法具有五个基本特征。

- (1) 有穷性 一个算法必须在有限次执行后完成。
- (2) 确定性 一个算法中的每一个步骤必须有明确的定义, 不能有语义不明确的地方。
- (3) 输入算法总是要施加到运算对象上, 提供运算对象的初始情况, 一个算法有 0 个或多个输入。
- (4) 输出 一个算法要有一个或多个输出。若无输出, 则无法知道结果。
- (5) 可行性 可行性是指所有待实现的运算必须是相当基本的, 至少在原则上人们可以用纸和笔做有限次操作即可完成。

实质上, 算法反映的是解决问题的思路。许多问题, 只要仔细分析对象数据, 就容易找到处理方法。

1.4.3 算法的表示

算法的表示方法很多, 主要有传统流程图、N-S 图、伪代码、自然语言和计算机程序语言等。这里重点介绍传统流程图和 N-S 图。

1. 传统流程图

用图形表示算法, 直观形象, 易于理解。流程图是用一些图框来表示各种操作。美国国家标准化协会 ANSI 规定了一些常用的流程图符号, 如图 1-1 所示。

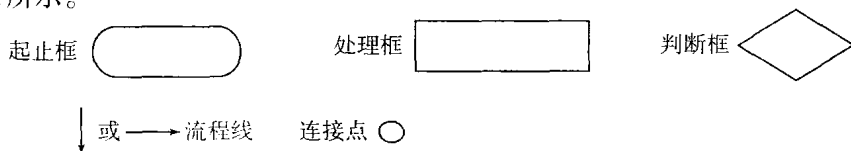


图 1-1 流程图符号