高等职业技术电子信息类专业教材

C 程序设计

田淑清　周海燕　赵重敏　林昱　编著

Publishing House of Electronics Industry

内 容 提 要

本书是高等职业技术电子信息类专业的教学用书。通过本书的学习，学生能够应用 C 语言进行初步的程序设计。

本书分为三个部分。第一部分介绍了三种数据类型的输入和输出，使之尽快用 C 语言编程上机实践。而后介绍函数初步知识，并运用函数来完成各种练习。第二部分引进了指针及数组，并进一步讨论了各种复杂的数据结构，列举了最常见的一些算法。第三部分介绍了“用户标识符的作用域”，“编译预处理”，“在终端上按格式进行输入和输出”等。

本书可作为大专院校电子信息类专业的教材，也可供其他相关专业学生及自学者参考。

图书在版编目(CIP)数据

C 程序设计/田淑清等编著. —北京：电子工业出版社，1998.9
高等职业技术电子信息类专业教材
ISBN 7-5053-4732-2

Ⅰ.C… Ⅱ.田… Ⅲ.C 语言-程序设计-高等教育-教材 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(98)第 17506 号

丛 书 名：高等职业技术电子信息类专业教材
书　　名：C 程序设计
编 著 者：田淑清　周海燕　赵重敏　林昱
责任编辑：吕　迈
排版制作：电子工业出版社计算机排版室
印 刷 者：
出版发行：电子工业出版社出版、发行　URL：http://www.phei.com.cn
　　北京市海淀区万寿路 173 信箱　邮编 100036　发行部电话 68214070
经　　销：各地新华书店经销
开　　本：787×1092　1/16　印张：20　字数：512 千字
版　　次：1998 年 9 月第 1 版　1998 年 9 月第 1 次印刷
书　　号：ISBN 7-5053-4732-2 / G·379
定　　价：　元

出 版 说 明

高等职业技术教育是现代教育的重要组成部分。近几年随着社会经济和科学技术的发展,已从客观上提出了发展高等职业技术教育的要求。高等职业技术教育在经历了认识定位和模式创新的阶段之后,开始进入课程建构和教材编写的新阶段。

在教育部职教司教材处的直接领导和电子工业出版社的积极组织下,三所积极发展高等职业技术教育的学校——北京联合大学、上海第二工业大学和深圳职业技术学院组建了教材编写领导小组。

三校教材编写领导小组经过多次研讨,认为目前没有能满足高等职业技术教育需要的现行教材,编写符合高等职业技术教育特点的教材已迫在眉捷。三校对电子信息类专业人才培养目标、职业定位以及电子信息类的内涵等问题达成共识,并将电子信息类教材作为首批开发的选题。

我们组织编写这套教材的原则是:充分探索高等职业教学特点,力图构筑以掌握基本概念、强化实际应用为重点,以获得职业技术所需的最基本、最适用的理论知识,以利于培养学生专业实践的适应能力和应变能力的新课程体系。

编写高等职业技术教育的教材是一个新课题,经验尚不足,希望全国电子信息类高职院校的师生积极提出批评建议,共同探索我国高等职业技术教育的特点和路子,不断提高教材的质量,最终形成电子信息类专业配套的高质量的教材。

三校教材编写领导小组

1998 年 4 月

三校教材编写领导小组

组长:牛梦成

组员:高　林　姚家伦　沈耀泉　吴金生

贡文清　朱懿心　戴士弘

目　录

第 1 章　C 程序设计的初步知识

1.1　简单 C 程序的组成和格式

本节将通过几个简单的程序例子，介绍 C 程序的一些基本概念，使读者对 C 语言程序的构成有一个初步的了解。

【例 1.1】在屏幕上输出一串字符：He is a student.

程序如下：

```
/* cex0101 */
#include <stdio.h>
main( )
{
        printf("He is a student.\n");          /* 输出字符串 */
}
```

以上程序运行后，会在屏幕上显示：

He is a student.

程序中的 main ( ) 表示主函数的起始行，main 是主函数名，C 语言规定必须用 main 作为主函数名。其后的一对圆括号中间可以是空的，但这一对圆括号不能省略。

每一个 C 程序都必须有一个而且只能有一个主函数。

函数中的语句放在一对花括号 {} 内，用这对花括号括起来的部分称为函数体。“{”表示函数体开始，“}”表示函数体结束。

函数体中可以有任意多个语句，本例的主函数体中只有一个输出语句——printf 语句，它调用了 C 语言标准库中的 printf 函数，此处，它的功能是把后面双引号中的内容原样输出到屏幕上(有关输出函数后面将详细讨论)。在汉字系统下，双引号中可以是汉字，这样就可以在屏幕上显示汉字了；双引号内最后的 \n 是换行符，在此它的作用是将双引号中的内容输出完毕后，立即把打印机头或屏幕光标换至下一行的起始位置，使下一次的输出从新的一行开始。

程序中的第 2 行 #include <stdio.h> 用于包含一些程序中所需要的信息，以便能成功地调用标准库中的输入、输出函数。它不是 C 程序中的语句，通常称它为命令行。

程序中用“/*”和“*/”符号括起来的一串字符是对程序的注释，他们必须成对出现；“/”和“*”之间不可以有空格，注释可以用西文，也可以用中文，注释可以出现在程序中任意合适的地方。虽然它对程序的运行不起作用，但是好的注释可以使人们在阅读程序时能较好地理解各段程序的功能、变量的含义。因此一个好的程序应该有详细的注释。

【例 1.2】给出圆的半径 r，计算圆周长 c 和圆面积 s。

已知求圆周长的公式为 $c = 2\pi r$，求圆面积公式为 $s=\pi r^2$。

```
/* cex0102.c */
#include <stdio.h>
```

```
main( )
{       int       r ;                          /* 定义变量 r,说明为整型 */
        float     c, s ;                       /* 定义变量 c 和 s,说明为实型 */
        r = 4;                                 /* 给 r 赋值 */
        c = 2 *3.141592 * r ;                  /* 计算圆周长,将值赋给 c */
        printf ("圆周长 c = %f\n",c);          /* 输出圆周长 */
        s = 3.141592 * r * r ;                 /* 计算圆面积,将值赋给 s */
        printf ("圆面积 s = %f\n",s);          /* 输出圆面积 */
}
```

以上程序运行后结果如下：

圆周长 c = 25.132736

圆面积 s = 50.265472

程序中的第 4 行和第 5 行是变量定义部分，在这里对程序中所用到的每一个变量进行定义并且说明其类型。在此 r 被说明为整型变量，因此赋给它的值应该是不带小数点的整型值；c 和 s 是实型变量，赋给它们的应该是带小数点的实型值。

第 6 行是赋值语句，这里将整型数值 4 赋给整型变量 r，使 r 的值为 4；第 7 行和第 9 行也是赋值语句，第 7 行首先计算表达式 2 * 3.14159 * r，然后把计算所得的值赋给变量 c，第 9 行则首先计算表达式 3.14159 * r * r，然后把计算所得的值赋给变量 s。

在第 8 行和第 10 行中一对双引号内的 %f 是输出的格式控制说明，用来指定输出时的数据类型和格式，%f 表示以标准格式输出一个实型数(即 c 和 s 的值)，当程序执行输出时，在 %f 所在的位置上将显示这个实数。

本小节给出了两个 C 程序，通常我们把由合法的 C 语句和程序行构成的 C 程序，称为 C 的源程序。每个 C 的源程序可包含任意多个函数(有关函数的初步知识将在第 5 章介绍)，但只能有一个 main 函数，程序总是从 main 函数开始执行。建议读者在教师或专业人员指导下，按原样逐个把以上两个源程序输入计算机，经过编译、连接，最后执行，看一看程序执行的结果，以便你对 C 程序有一个直接的感性认识。在此顺便提醒读者：要想掌握 C 语言，唯一的途径就是要大量地进行上机练习。

注意：C 语言程序有比较自由的书写格式，但是过于“自由”的程序书写格式，往往使人们很难读懂程序，因此要求读者在学习编程的过程中，参考本书例题程序的书写格式来写自己的程序，这一要求将贯穿整个学习过程。

1.2 十进制整型数和实型数

1.2.1 常量

在 C 语言中，把在程序运行过程中其值不能改变的量称为常量，如例 1.2 中的 4、2、3.141592等。常量分为不同的类型，有整型常量、实型常量、字符常量和字符串常量。为了便于初学的读者尽快地进入 C 程序设计，本节只讨论整型和实型常量，其它内容将在后续章节中陆续介绍。

整型常量和实型常量也称数值型常量，它们有正值和负值的区分。整型常量必须不带小数点，如 3、－3、0 等；实型常量通常带小数点，如 3.14159、－1.23、0.0 等。由此可见，常量的

类型从字面形式即可区分，C 编译程序就是以此来确定数值常量的类型的。

1.2.2 十进制整型常量

整型常量也称整常数，在 C 语言中，整型常量可以用十进制的形式来表示，如 123、－345、0 等，也可以用八进制、十六进制的形式来表示。在这里只介绍用十进制形式表示的整常量。有关八进制，十六进制形式的整数将在第 17 章中介绍。

整型常量又有短整型(short int)、整型(int)、长整型(long int)和无符号型(unsinged)的区分。在第 1 至第 16 章我们将只涉及整型常量，它的取值应在 －32768～32767 范围内。其余的内容可查阅第 17 章。

1.2.3 浮点常量

浮点常量在 C 语言中又称实型数。浮点常量有两种表示形式：

(1)带小数点形式，即日常在数学中采用的实数形式，它由数字和小数点组成(注意：必须要有小数点)，如 0.123、.123、123.、0.0 等都是合法的浮点常量。

(2)指数形式。这种形式类似数学中的指数形式。在数学中，一个数可以用幂的形式来表示，如 345.89 可以表示为 34589×10^{-2}、3458.9×10^{-1}、3.4589×10^{2} 等。在 C 语言中，则以“e”或“E”来表示以 10 为底的幂数。如以上列举的 345.89 可依次写成 34589e－2、3458.9E－1、3.4589e2。但应注意的是，字母 e 或 E 之前必须要有数字，且 e 或 E 后面的指数必须为整数。如果写成 e3、.5e3.6、.e3、e 等都是不合法的指数形式。另外，字母 e(或 E)的前后与数字之间不得插入空格。

C 语言中的浮点常量都被识别为是双精度类型(double)的，也就是说(视机器的不同)它们可有 15 至 16 位的有效位。

1.2.4 用定义一个符号名的方法来代表一个常量

可以用一个符号名来代表一个常量，但是这个符号名必须在程序中进行特别的“指定”。

【例 1.3】计算圆周长和圆面积。

```
/*    cex0103.c    */
#include              〈stdio.h〉
#define       PI              3.141592                    /* 定义符号名 PI 为 3.14159 */
main( )
{       int             r;
        float           c, s;
        r = 4;
        c = 2 * PI * r;
        printf ("c = %f\n",c);
        s = PI * r * r;
        printf("s = %f\n",s);
}
```

程序运行结果如下：

```
c = 25.132736
```

s = 50.265472

程序中用 #define 命令行(注意:不是语句)定义 PI 代表一串字符 3.14159,在对程序进行编译时,凡本程序中出现 PI 的地方,编译程序均用 3.14159 这一串字符来替换,在这里,PI 是一个用户自己选择的符号名,本程序中,可以把 PI 视为 3.14159 的替身。为了使之比较醒目,这种符号名通常采用大写字母。另外必须注意,在 #define 命令行的最后不得加分号,有关 #define 命令行的作用,将在第 14 章中介绍,读者可以先按上述方法简单地使用。在很多 C 语言书中把 #define 命令行中定义的符号名也称为符号常量。

1.3 标识符

在 C 语言中,用来标识变量名、符号名、函数名和后面将要学习的数组名、文件名以及一些具有专门含义的名字称为标识符。C 语言中,合法的标识符只能由字母、数字和下划线组成,并且第一个字符必须为字母或下划线。下面的标识符都是合法的标识符:

sum　　ave　　_r　　_bas　　a104　　c101

以下都是非法的标识符:

m.d　　110a　　k+5a　　$a3

在 C 语言的标识符中,大写字母和小写字母被认为是两个不同的字符,因此 ave 和 Ave 是两个不同的标识符。

对于标识符的长度(即一个标识符允许的字符个数),一般的计算机系统规定取前 8 个字符有效,如果长于 8 个字符,多余的字符将不被识别。如 class1110 和 class1112 在计算机系统内则被认为是相同的标识符——class111。C 语言的标识符可以分为以下三类。

1.3.1 关键字

C 语言规定了一批标识符,它们都代表着固定的含义,不能另作它用。例如,用来说明变量类型的标识符 int、float 以及 if 语句中的 if、else 等都已有专门的用途,它们不能再用作变量名或函数名。C 语言中的关键字见附录 A。

1.3.2 预定义标识符

预定义标识符在 C 语言中也都有特定的含义,如 C 语言提供的库函数的名字(如 printf)和预编译处理命令(如 define)等等。C 语言语法允许用户把这类标识符另作它用,但这将使这些标识符失去系统规定的原意。鉴于目前各种计算机系统的 C 语言都一致把这类标识符作为固定的库函数名或预编译处理中的专门命令使用,因此为了避免误解,建议用户不要把这些预定义标识符另作它用,否则会带来不必要的麻烦。

1.3.3 用户标识符

由用户根据需要定义的标识符称用户标识符。一般用来给变量、函数、数组或文件等命名。

程序中使用的用户标识符除要遵循起名规则外,还应注意做到“见名知义”,即,选有含义的英文单词,如 name、day、month、class、count 或汉语拼音,以增加程序的可读性。

如果用户标识符与关键字相同,程序在编译时将给出出错信息;如果与预定义标识符相

同,系统给予承认,并不报错,只是该预定义标识符将失去原定含义,代之以用户确认的含义。

1.4 整型变量和实型变量

1.4.1 变量

程序中的变量由用户取名,如例 1.2 中的 r、c 和 s 就是由用户定义的变量名。程序中所用到的每一个变量都应该有一个名字作为标识,称为"用户标识符"。变量名的命名规则应遵守标识符命名规则。

程序中,一个变量实质上是代表了内存中的某个存储单元,所以,所谓给一个变量赋值,实质上就是把数据存入该变量所代表的内存单元中。

C 语言规定,程序中所要用到的变量必须先定义后使用,因为只有经过定义的变量,系统在编译过程中才能为其开辟一定数量的存储单元。在这里应该注意的是变量的"名"和变量的"值"是两个不同的概念,"名"是指内存单元的标志,即某个地址上的存储单元;"值"是指内存单元中的内容,即存放在该存储单元中的数据。

在 C 程序中,对变量的定义通常放在函数的开头部分(也可以放在函数的外部或复合语句的开头),但无论在何处定义,只有从定义位置开始被定义的变量才具有实际意义。

像常量一样,变量也有类型的区分,如整型变量、实型变量、字符型变量等。C 语言在定义变量的同时说明了其类型,系统在编译时就能根据其类型为其分配相应类型的存储单元。

1.4.2 整型变量

整型变量可以分为基本型、短整型、长整型和无符号型四种。为了使读者能尽快地使用变量进行初步的程序设计,本节将只介绍基本类型的整型变量,其它的内容将在第 17 章详细地介绍。

基本型的整型变量用类型名关键字 int 进行定义,整型变量定义的形式如下:

```
int     w ;          /* 定义变量 w 为整型 */
int     a,   b;      /* 定义变量 a、b 为整型 */
```

一个定义语句必须以一个";"号结束。在一个定义语句中也可以同时定义多个变量,变量之间用逗号隔开。

一般微机为基本型整型变量开辟 2 个字节(16 个二进制位)的内存单元,并按整型数的存储方式存放数据。整型的变量只能存放整型数值,数值的范围是:$-32768 \sim 32767$(即,$-(2^{15}) \sim 2^{15}-1$)。

1.4.3 浮点型变量

C 语言中浮点型变量分为单精度型和双精度型两类,分别用类型名 float 和 double 进行定义。定义的形式如下:

```
float     x,   y ;
double    w,   s ;
```

在一般计算机系统中,为 float 类型的变量分配 4 个字节的存储单元,为 double 类型的变量分配 8 个字节的存储单元,并按实型数的存储方式存放数据。说明为实型的变量只能存放实型数值;不能用整型变量存放一个实型数,也不能用实型变量存放一个整型数。