第一章 概 述

C 语言是目前国际上广泛流行的一种结构化的程序设计语言，它不仅适合于开发系统软件，而且也适合于开发应用软件。因此，它受到了广大的程序设计者的欢迎。

本章简要介绍 C 语言的发展和特点，C 语言源程序的结构以及它的执行过程。

第一节 C 语言简介

本节将对 C 语言进行简单的介绍，内容包括 C 语言的产生和发展，C 语言的特点。

一、C 语言的产生和发展

对 C 语言的产生追本求源，可追溯到 ALGOL(Algorithmic Language) 语言。1960 年出现了 ALGOL 60，它是一种面向问题的高级语言。远离硬件，不适于开发系统软件。1963 年，英国剑桥大学推出了 CPL(Combined Programming Language)语言，CPL 语言比 ALGOL 语言接近硬件一些，但规模较大，难以实现。 1969 年，剑桥大学的 M. 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 K. Thompson 在 BCPL 的基础上又作了简化，设计出了更接近硬件的 B 语言。并用 B 语言编写了在 PDP-7 计算机运行的早期的 UNIX 操作系统。由于 BCPL 语言和 B 语言无数据类型，过于简单，功能有限，1972 年至 1973 年期间，贝尔实验室的 D. M. Ritchie 在保留 BCPL 语言和 B 语言优点的基础上，设计了 C 语言。1973 年，用 C 语言改写了 UNIX 操作系统，这就是 UNIX 第五版。最初的 C 语言只是为描述 UNIX 操作系统的工作语言，依附于 UNIX，主要在贝尔实验室内部使用。后来 C 语言作了多次改进，到 1975 年，UNIX 第六版的公布，使 C 语言受到人们普遍的注意。

UNIX 操作系统的广泛使用，促进了 C 语言的发展和普及，而 C 语言的发展和普及又促进 UNIX 的推广。1978 年出现了独立于 UNIX 和 PDP 计算机的 C 语言，C 语言迅速地被移植到大、中、小和微型计算机上。当年，B. W. Kernighan 和 D. M. Ritchie 以 UNIX 第七版的 C 编译程序为基础，合作编写了《The C Programming Language》。K&R 的这本名著给出了 C 语言的经典定义，故把它作为 C 语言的标准。此后，各种 C 语言编译系统纷纷出现。1983 年，美国国家标准协会(ANSI)根据各种版本对 C 语言的扩充和发展，制订了新的标准。称为 ANSI C。ANSI C 比原来的标准 C 有了很大的

发展。K&R 也以新标准改写了他们的著作。现在一般把 ANSI C 称为新标准，把 K&R 在他们的著作中介绍的称为旧标准。本教材介绍的内容是以 ANSI C 为基础。

目前，微型计算机上广泛使用的 C 语言的版本为 Turbo C、Quick C、Microsoft C/C++、Visual C/C++ 等，Turbo C 2.0 完全支持新标准，提供集成开发环境。本教材以此作为实践环境。本教材在附录 E 中，简单介绍了在 Turbo C 2.0 的集成环境下上机的操作步骤。

二、C 语言的特点

C 语言之所以能成为目前国际上广泛流行的程序设计语言，是由它的特点决定的。C 语言的主要特点如下：

(1)C 语言简洁。C 语言和 PASCAL 语言都是在 ALGOL 语言的基础上发展而来的，C 语言和 PASCAL 语言相比，它要比 PASCAL 语言表达简洁。如 C 语言中，定义 i 为整型变量表达形式为 int i，在 PASCAL 语言中，表达同样的问题的形式为 VAR i:INTEGER。C 语言和 FORTRAN 语言相比，FORTRAN 语言书写程序有格式约束，而 C 语言程序书写形式自由，使用方便、灵活。再有 C 语言的构成也简洁，ANSI C 有 32 个关键字，9 种控制语句，但能完成程序设计中的所有控制功能和其他功能。

(2)C 语言提供了类型丰富和使用灵活的运算符。ANSI C 提供了 34 种运算符，这些运算符不仅可以完成其他高级语言所具有的算术运算和逻辑运算，而且可以完成复合运算和位运算等。其中有些运算是其他高级语言所难以实现的。C 语言的运算符丰富，同一种运算可用不同的运算符实现，使表达式简练、多样、灵活、实用。如使变量 i 的值增 1，可以使用的表达式有，i++，++i，i+=1，i=i+1 等。另外，C 语言的运算符功能强大，在很多场合，可以使用一个表达式达到其他高级语言几条语句的功能。

(3)C 语言数据类型丰富，具有现代程序设计语言的各种数据类型。C 语言中不仅有简单的数据类型，而且允许用户构造数据类型，能够实现各种复杂的数据结构，完成各种问题的数据描述。尤其是 C 语言中的指针类型，独具特色，可用它指向各种数据，对数据进行高效处理。C 语言中使用数据时不但要作数据类型的描述，而且还要考虑它们的存储属性。

(4)C 语言是一种结构化的程序设计语言。结构化的程序可以用顺序、选择和循环三种基本结构组成。C 语言具有如 if～else、switch～case、for、while、do～while 等结构化的语句，非常便于程序设计者采用自顶向下、逐步细化的结构化程序设计技术。使用 C 语言编制程序具有容易理解，便于维护等优点。

(5)C 语言是一种模块化程序设计语言。C 语言程序是由一系列函数所组成，这种结构便于把大型程序划分为若干个相对独立的模块，每个模块完成一定的功能，模块之间通过函数调用来实现相互连接。这样，就为集体共同开发一个大型的软件提供了方便。

(6)C 语言虽是高级语言，但它具备许多汇编语言的功能。C 语言能直接访问物理

地址，能对位进行操作，比较接近硬件和系统，能实现汇编语言的许多功能。所以，它既适合用来编写系统程序，也适合用来编写应用程序。既是系统的描述语言，又是通用的程序设计语言。

(7)C 语言的程序中允许使用编译预处理命令行，能够进行字符串或特定参数的宏定义，以及实现对外部文本文件的操作等。这些功能提高了程序的执行效率。

(8)用 C 语言编制的程序移植性好。C 语言中没有依赖于硬件的输入/输出语句，程序的输入/输出是靠调用标准库函数中的输入和输出函数实现的。这样，就使 C 语言程序本身不依赖于硬件系统，便于在不同的机器系统之间的移植。

以上只是 C 语言的一般特点，这些特点以及 C 语言的其他特点还要在以后的学习中进一步理解和体会。由于 C 语言的优点突出，深受程序设计者的偏爱。希望读者努力学习,尽快掌握 C 语言，灵活运用它，开发出高水平的软件。

第二节　简单的 C 语言程序的组成和格式

不同的高级语言程序的组成和格式是不同的，本节先给出几个 C 语言程序的例子,使读者对 C 语言程序的结构有一个初步的认识。

[例 1.1]　编写在屏幕上显示字符串“How do you do!”的程序

程序如下：

```
#include <stdio.h>
main()
{
    printf("How do you do!  \n");
}
```

程序运行情况如下：

```
How do you do!
```

以上给出的程序运行情况是在显示器的屏幕上显示的字符串。该程序的组成和格式介绍如下：

(1)该程序由一个主函数 main()组成。C 语言中的任何一个程序都必须有此函数。函数名必须是 main 且其后的一对括号不能少，括号内可以是空的。

(2)一对花括号{　}所括起的部分是 main 函数的函数体。“{”表示函数体的开始，“}”表示函数体的结束。函数体内可以有多个语句，本例中只有一个输出语句。

(3)printf()是系统提供的标准库函数中的输出函数。C 语言程序的输出是由输出函数来完成的。“How do you do!”是要输出的内容。“\n”表示换行。它是由“\”和“n”两个字符构成，属转义字符。在 printf()函数中输出的内容和转义字符都要用双引号括起来。有关 printf()函数和转义字符在后续章节中做详细介绍。

(4)在 C 语言中，分号；作为语句的结束符。如本例中在 printf()函数后面加了一个“;”.printf()变成了语句。

(5) #include 是预编译程序命令，它把头文件“ stdio. h”的内容展开在 #include <stdio.h>所在的行位置处。所以，在每个使用标准库函数的程序中，都必须带有 #include 命令行。在该命令行中使用的一对尖括号 < >，也可使用一对双引号。即 #include "stdio. h"。

[例 1.2] 当给出圆的半径时，编写计算圆的周长的程序

分析：由数学知识可知：圆的周长 $c=2\pi r$。程序中先给变量 r 赋值，再用此公式进行计算,最后,输出结果 c。

程序如下：

```
#include "stdio.h"
main()
{int r;                          /*定义变量 r,说明其为整型*/
 float c;                        /*定义变量 c,说明它为实型*/
 r=8;                          /*   给 r 赋值*/
 c=2*3.14159*r;                  /*计算圆周长，将值赋给 c*/
 printf("c=%f\n",c);             /* 输出圆周长*/
}
```

程序运行情况如下：

```
c=50.265442
```

以上程序中的第三行和第四行是变量的定义部分，在这里，对程序中用到的每一个变量进行定义并且说明其类型。 r 被说明为整型变量，所以，赋给 r 的值为整型值。c 是实型变量，赋给它的值为带小数的实型值。

第五行是赋值语句，把整型数据 8 赋给了整型变量 r,使 r 的值为 8。第六行也是赋值语句,在这里,先计算表达式 2*3.14159*r ,然后把计算所得到的值赋给变量 c.

第七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据，当输出时，在 %f 所在的位置上显示这个实数。此行中一对双引号内的 c= 是原样照印的字符串，双引号外的 c 是将要输出的变量值的变量名。

[例 1.3] 编写求两个整数中较小数的程序

分析：本题目编写两个函数。即 min()函数和 main()函数，min()函数的功能是比较两个整数的大小，并返回其中较小的数。 main()函数用来输入这两个整数，调用 min()函数及输出结果。

程序如下：

```
#include "stdio.h"
main()
```

```
{   int a,b,c;                          /* 定义变量 */
    printf("Input a and b:");           /* 输出提示信息 */
    scanf("%d%d",&a,&b);                /* 输入变量 a 和 b的值 */
    c=min(a,b);                         /* 调用 min 函数，并把函数的返回值赋给 c */
    printf("min=%d\n",c);               /* 输出较小的数 */
}
int min(int x,int y)                    /* 定义 min 函数，函数值为整型 */
{   int z;                              /* 定义变量 z */
    if(x<y) z=x;                        /* 找两个数中较小的数 */
    else  z=y;
    return z;                           /* 将 z 的值返回，通过 min 带回到调用处 */
}
```

程序运行情况如下：

```
Input a and b:3 5
min=3
```

以上程序不仅包括主函数 main()，还包括 min()函数，min()函数是用来求两个整数中较小值的。有关函数的问题将在第七章做介绍。在这里，只是说明一个 C 语言程序可以由多个函数构成并给出函数的定义形式。

主函数中的“scanf("%d%d",&a,&b);”是输入语句，scanf()是格式化输入函数，它是一个由系统提供的标准库函数，其后括号内双引号括起的 %d 为格式说明符，%d 表示输入带符号的十进制整型数，执行该语句时，变量 a 和 b 的值必须以十进制整数形式由键盘输入。

由以上三个例子可以看出：

(1)C 语言程序由一个或多个函数组成。其中必须有一个名为 main 的主函数。无论 main 函数在整个程序中的位置如何，程序的执行总是从主函数开始的。主函数中的所有语句执行完，则程序执行结束。其他函数都是在执行 main() 函数时，通过调用或嵌套调用而得以执行。这些被调用的函数可以是系统提供的库函数，也可以是程序设计者根据需要自己编制的函数。

(2)C 语言程序中定义的函数包括函数说明和函数体两部分。函数说明指明函数类型、函数的属性、函数名、函数参数名和形式参数类型。函数名是一个标识符（标识符的规定在第二章介绍），可以用除关键字和 C 语言库函数名以外的名字。函数类型是指函数返回值的类型，可以是 C语言中任何一种简单或构造数据类型。函数属性是指函数的内部或外部属性。函数参数名是指形式参数名，即可以向函数传递的变量代表名，也是标识符。形式参数类型就是形式参数的数据类型。在 C 语言中，函数的形式参数类型可以放在函数说明部分的圆括号外面，也可以放在圆括号的里面。函数体则是函数说明部分下面的从“{”开始，到“}”结束的部分，函数体包括对数据对象描述的变

量的定义和对算法描述的语句序列。对函数的详细介绍将在第七章进行。

(3)习惯用小写英文字母书写 C程序，大写英文字母作为常量的宏定义和其他特殊用途。

(4)C 程序书写格式自由，一行内可以写多条语句，一条语句也可分写在多行上。但考虑程序的清晰度，最好在一行内写一条语句。

(5)每个语句和变量的定义的最后必须有一个分号，分号是语句的组成部分。

(6)函数定义之外可以包含说明部分，此说明部分可称为外部说明，它可包括预编译命令（如上面例子中的 #include)和外部变量的说明等。

(7）函数中允许有一对以上的花括号。这是由于 C 语言中有些控制语句如 if、while、do、for、else、switch 等经常是由若干语句组成的复合语句，复合语句需要用花括号括起来。这样一个 C语言程序中可能出现多层花括号，最外层的花括号表示函数体的开始和结束。内层的花括号表示复合语句的开始和结束。实际上，花括号也就表示了程序结构层次的范围。

(8)C 语言程序中也可以使用注释。注释部分的格式为：

/* 注释内容 */

如 :例 1.3 中所有语句都加了注释，注释在程序的编译过程中不产生任何执行代码，只是在程序中起到解释、说明的作用。

(9)C 语言中没有输入/输出语句，程序的输入和输出是通过调用标准库函数中的输入函数、输出函数实现的。

第三节　C 语言源程序的执行过程

我们把用 C 语言编写的程序，称为 C 语言源程序。一个 C 语言源程序要在计算机上执行,也和那些采用编译方式的高级语言一样,经过编辑、编译、连接、运行四个步骤。此过程如图 1-1 所示。

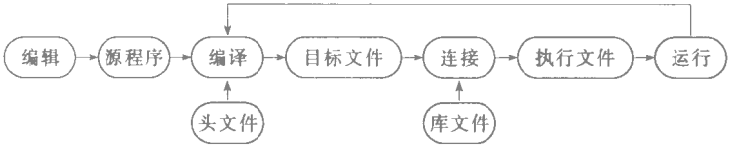

图 1-1　C 语言源程序的运行过程

1.编辑

程序设计者首先要使用系统提供的编辑程序（又称编辑器），把自己编制的 C语言的源程序输入到计算机，并以文本文件的形式存储在磁盘上。我们把这种文件称为源文件。源文件的名字由程序设计者选定，其扩展名为“.c”例如 file1.c.

一个 C语言的源文件是一个编译单位，一个大的 C 语言源程序可划分为若干个模

块，每个模块可以由不同的人来编写，每个模块建立一个源文件。这样，一个大的 C 语言的源程序可由多个源文件组成。

用于建立源文件的编辑程序种类较多，如 MS-DOS 下的 EDLIN，WordStar 等。一般使用集成的编辑软件，这种软件集编辑、编译、连接、运行为一体，使用起来非常方便。如 Turbo C。

2. 编译

在源文件编译前，要对其再次进行检查。如在 MS-DOS 下用 type 命令查看源文件的内容。发现问题后，重新启动编辑程序对源文件进行修改。正确后就可以对源文件进行编译。

编译是由系统提供的编译程序（又称编译器）完成的。编译命令随系统不同而不同，具体操作时可参考相应的系统手册。正确使用编译命令实现编译过程。

系统对源文件编译前，先进行预处理。编译过程主要进行词法分析和语法分析。当发现错误时，就在显示器上列出错误的位置和种类。此时，要再次使用编辑程序对源文件进行修改。修改后，再次进行编译。

正确的源文件编译后生成目标文件并存储于文件系统中。目标文件带有规定的扩展名。如在 MS-DOS 中，扩展名为“.obj”.

编译也可使用集成的编译软件。如 Turbo C。在 Turbo C 的集成环境下，使用编辑界面中的“Compile”菜单中的“Compile”命令进行编译。也可以选择“Compile”菜单中的“Make”命令进行编译，它能直接生成可执行文件。在编译的过程中，系统若发现源程序有错误，就给出错误的参考信息。此时就要对源程序进行修改，修改后再进行编译。

3. 连接

编译后形成的目标代码程序不能在机器上直接执行，还必须使用连接程序（又称连接器），将目标代码程序和系统提供的库函数以及需要用到的其他目标文件连接起来，才能生成可执行的文件。生成的可执行文件存储在文件系统中。 MS-DOS 下可执行的文件的扩展名为“.exe”。

4. 运行

可执行文件生成后，就可以执行它。若在 MS-DOS 下，直接输入可执行的文件名即可运行。若运行结果达到要求，则完成任务。否则，就需要检查修改源程序，重复以上的步骤，直至达到要求为止。

习　　题

1.1　C 语言有哪些主要的特点？

1.2　C 语言的程序由什么构成的？

1.3　在 C 语言中被定义的函数包括哪两部分？

1.4　C 语言的程序的书写有何特点？

1.5 若在一个 C 语言的程序中有多个函数，其中必须有一个且只有一个函数是主函数，主函数的名是什么？程序从哪个函数开始执行？其他函数是如何执行的？

1.6 注释在程序中起什么作用？请给例 1.1 程序加上注释。

1.7 C 语言中数据的输入和输出是如何实现的？

1.8 一个 C 语言的程序在机器上运行要经过哪些步骤？

1.9 在机器上运行例 1.1、例 1.2、例 1.3 中的程序，熟悉实验环境。

1.10 模仿例 1.1，编写程序，输出字符串“We are studing C。”

1.11 模仿例 1.2，编写程序，求两个整数之和。

1.12 模仿例 1.2，编写程序，求三个实型数的积。

1.13 模仿例 1.2，编写程序，已知正方体的边长为 5cm，求正方体体积。

1.14 模仿例 1.2，编写程序，已知半径为 4 cm，高为 8 cm，计算圆柱体体积。

1.15 模仿例 1.2，编写程序，已知三角形的三条边分别为 3 cm，4 cm，5 cm，求三角形的面积。

1.16 模仿例 1.3，编写程序，找出两个数中的较大的数。

第二章　基本数据类型和基本运算

一个计算机程序是用来描述对象的属性和行为的，属性的值是用数据来表示的，对象的行为是用来处理数据的。因此，数据是程序加工操作的素材。我们把数据在加工中的特征称为数据类型。

C 语言中的基本数据类型,有数值型(包括整型和实型）、字符型和枚举型。为了描述更复杂的对象,C 语言还设有一些更复杂的数据类型,具体如图 2-1 所示。

C 语言中,数据有常量和变量之分,它们分别属于以上类型。

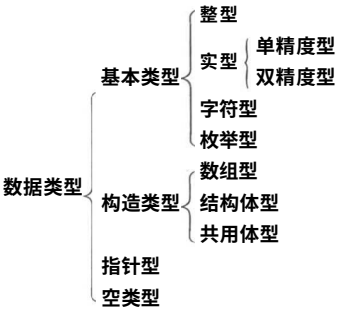

图 2-1　C 语言的数据类型

第一节　常　量

在程序运行过程中，其值不能被改变的量称为常量。在 C 语言中,常量有不同的类型,有数值型常量,字符常量和字符串常量。其中数值常量又分为整型常量(整数)和实型常量（实数）。常量的类型从字面形式可以判断。如 10、0、-28 为整数,3.14、-6.36 为实数,'a'、'x' 为字符常量,"book" 为字符串常量。

一、数值常量

1.整型常量

C 语言中，整型常量可以用以下三种形式表示：

①十进制整数。如 135,0,-246。

②八进制整数。以前导 0 开头的数是八进制数，如 0135 表示八进制数 135,即 $(135)_8 = 1 * 8^2 + 3 * 8^1 + 5 * 8^0 = (93)_{10}$

③十六进制整数。以前导 0x 开头的数是十六进制数，如 0x135 表示十六进制数 135,即 $(135)_{16} = 1 * 16^2 + 3 * 16^1 + 5 * 16^0 = (309)_{10}$ -0x15 表示十进制数 -21。

整型常量又有基本型、短整型、长整型和无符号整型之分,若要表示一个长整型常量，则应该在一个整型常量后面加一个字母后缀 l(小写的字母)或 L,如 123L、345L、0L、654321L 等，这些常量在内存中占四个字节。基本型和短整型的常量在内存中都占

两个字节。注意，一个足够大的数，如 123456，虽然其面值是在长整型范围内，但由于数字后面未加后缀 L，因此并不能代表一个长整型数。

无论是短整型还是长整型，都被识别为有符号整数，无符号整数在数字的末尾应该加上字母后缀 u 或 U，若是长整型无符号整数常量，则应该加后缀 lu 或 LU；短整型无符号常量的取值应在 0～65535 范围内，长整型无符号常量的取值则在 0～4294967295 范围内。注意无符号常量不能表示成小于 0 的负数，例如 -100U 是不合法的。

2.实型常量

实数又称浮点数。有两种表示形式：

①十进制小数形式。由数字和小数点组成（注意必须有小数点）。如 0.123、.123、123.、123.0、0.0 等都是十进制小数形式。

②指数形式。这种形式类似数学中的指数形式。在数学中，1 234.567 可以表示为 $1.234\ 567 * 10^3$，还可以表示为 $12\ 345.67 * 10^{-1}$ 等形式。在 C 语言中则以"e"或"E"后跟一个整数来表示以 10 为底的幂数。1 234.567 可以表示为 1.234 567e3、12.345 67e2、123 456 7e-5 等。其中 1.234567e3 称为"规范化的指数形式"，即在字母 e（或 E）之前尾数的小数点前有且只有一个非零的数字，实数按指数形式输出时，将输出规范化的指数形式。C 语言的语法规定，字母 e（或 E）之前必须要有数字，e（或 E）之后必须为整数。如 3e、e3、3e4.5、e、等都是不合法的指数形式。

二、字符常量

C 的字符常量是用单引号括起来的单个字符。如 'A'、'a'、'B'、'+'、'!'、'$' 等都是字符常量。C 语言中，字符常量具有数值即 ASCII 码，字符常量的 ASCII 码见附录 C。例如，'A' 的 ASCII 码是 65，'a' 的 ASCII 码是 97，'A' 和 'a' 是不同的字符常量。

三、字符串常量

字符常量是单引号括起来的单个字符。如果是多个字符组成的字符序列（即字符串）如何表示呢？在 C 语言中，字符串常量是用双引号括起来的字符序列。如"Hello"、"CHINA"、"a"、"$1.23" 都是字符串常量。可以输出一个字符串，如：

```
printf("CHINA \n");
```

或 `printf("%s \n","CHINA");`

注意：字符常量和字符串常量不要混淆。'a' 是字符常量，而"a"是字符串常量。

在内存中，字符 'a' 占一个字节，而字符串"a"占两个字节。如图 2-2 所示。

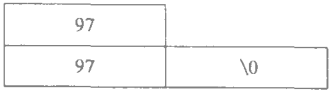

图 2-2 字符常量和字符串存储示意图

也就是说，在每个字符串的结尾加一个字符串结束标志，以便系统判断字符串是否

结束 C规定以字符‘\0’作为字符串结束标志。‘\0’是ASCII码为0的字符,是“空操作字符”。因此字符串“HELLO!”在内存中占7个字节。如图2-3所示。

H	E	L	L	O	!	\0

图2-3 字符串在内存中存储的示意图

四、转义字符常量

C还允许用一种特殊形式的字符常量，就是以一个‘\’开头的字符序列。例如‘\n’代表一个“换行”符。这类字符称为“转义字符”。意思是将反斜杠(\)后面的字符转换成另外的意义。常用的转义字符如表2-1所示。

表2-1 C语言中常用的转义字符

字符形式	含义	ASCII
\n	换行	10
\t	横向跳格	9
\b	退格	8
\r	回车	13
\f	走纸换页	12
\\	反斜杠字符	92
\'	单引号字符	39
\"	双引号字符	34
\ddd	1~3位八进制码字符	
\xhh	1~2位十六进制码字符	

其中‘\ddd’表示ASCII码值是八进制的ddd的字符，例如‘\101’表示大写字每‘A’，同样‘\xhh’表示ASCII码值是十六进制的hh的字符,例如‘\x41’表示大写字每‘A’。

五、符号常量

在C语言程序中，可以用一个符号名来代表一个常量，这个符号名要进行定义。请看下面的例子：

[例2.1] 计算圆面积、圆周长。

```
#include "stdio.h"
#define  PI  3.14
main()
{  float  r,s,c;
   r=2.0;
   s=PI*r*r;
   c=2*PI*r;
```

```
    printf("r= %f,s= %f,c= %f \n",r,s,c);
}
```

程序运行情况如下：

```
r= 2.000 000,s= 12.560 000,c= 12.560 000
```

程序中用#define 命令定义 PI 代表一串字符 3.14 ，在对程序进行编译预处理时，凡是程序中出现 PI 的地方，均替换为 3.14 这一字符串，习惯上用大写字母表示。注意 #define 命令必须用 #开头，命令行的最后不得加分号。通常把#define 命令行中定义的符号名称为符号常量。 #define 命令的详细用法将在第十章中介绍。

第二节　变量及其数据类型

在程序运行的过程中其值可以改变的量称为变量。 每一个变量都应该有一个名字，名字实际上是个标识符，在内存中占据一定的存储单元，在该存储单元中存放变量的值。请注意区分变量名和变量值。如图示变量名为 b,变量值为 30。如图 2-4 所示。

图 2-4　变量名与变量值示意图

一、标识符

在 C 语言中，用来标识符号常量、变量名、数组名、函数名、文件名等有效字符序列称为标识符。合法的标识符由字母、数字和下划线组成，并且第一个字符必须为字母或下划线。下面列出的是合法的标识符：

PI、area、circle 、day、month、student_num、Lesson_1、_1

下面是不合法的标识符：

123x、m－n 4th、John&Mike

在 C语言中，大写字母和小写字母被认为是两个不同的字符，因此 Sum 和 sum 代表不同的标识符。一般变量名用小写字母表示。

一般的计算机系统规定标识符的长度取前 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 lesson_10 、lesson_11 将会被认为是同一个标识符。 Turbo_C 则允许 32 个字符。为了方便起见，建议长度不要超过 8 个字符。而且应该做到“见名知意”，即选择有含义的英文单词（或缩写）作标识符。

二、变量的数据类型

像常量一样，变量也有类型之分，如整型变量、实型变量、字符型变量。在 C语言中，要求对变量要“先定义，后使用”。变量定义的一般形式如下：

数据类型标识符　变量名［,变量名,…］;

在这里，数据类型标识符指的是变量的数据类型，如 int、float、char 等,以后各章节中简称数据类型。C 语言在定义变量的同时说明该变量的类型，编译时据此定义及其

类型为它们分配相应大小的存储空间。

1. 整型变量

(1)整型变量的分类

整型变量分为带符号型和无符号型，无论带符号型和无符号型按所占存储空间大小又分为基本型、短整型和长整型。不同的计算机对这几类整型数所占存储空间的字节数和数值范围有不同规定。表 2-2 列出了微型机(IBM PC 机）中这几类整型数所占用的字节数和数值范围。表中方括号中的单词可写也可不写，单词的先后顺序无关紧要 。

表 2-2 整型变量类型

类型名称	占用的字节数	数值范围
[signed]int	2	-32 768～32 767
[signed]short[int]	2	-32 768～32 767
[signed]long[int]	4	-2 147 483 648～2 147 483 647
unsigned [int]	2	0～65 535
[unsigned] short [int]	2	0～65 535
[unsigned] long [int]	4	0～4 294 967 295

(2)整型数据在内存中的存储形式

数据在内存中以二进制形式存放的。在 C 语言中,一个 int 整数通常用两个字节存放，其中最高位（最左边的一位）用来存放整数的符号，对于正整数，符号位为 0,对于负整数,符号位为 1。实际上，数值是以补码表示的，要计算一个数的补码，就要涉及到原码和反码的概念。对于正整数的原码、反码和补码都相等，符号位（最高位）为 0,数值位(低 15 位）为该数的二进制形式（不够 15 位的补足 15 位）。例如,[+5]原码 =[+5]反码 =[+5]补码，其在内存中的存储形式如图 2-5 所示。

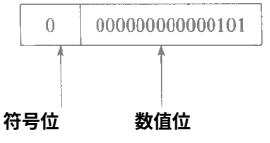

图 2-5 正数在内存中存储形式示意图

负数的原码形式是符号位为 1，数值位为其二进制形式：

例如,[-5]原码 =1000000000000101

负数的反码在其原码的基础上求得。具体方法是符号位仍为 1,数值位变反,即把 1 转换为 0,把 0 转换为 1：

例如,[-5]反码 =1111111111111010

负数的补码在其反码最低位加 1

[-5]补码 =1111111111111011

对于无符号数，最高位不再存储符号位，16 个二进制位全部用来存放整数，因此，无符号数不会是负的，最小为 0,最大为 16 位上全是 1,它代表整数 65 535。

2. 实型变量

C 语言中实型变量分为单精度型和双精度型两种，分别用类型名 float 和 double 进行定义。如：

```
float a;
double b;
```

不同的系统为它们分配的字节数不同，这两种类型变量分配的存储空间大小、表示数的范围、有效数字位数也不同，微型机（IBM PC 机）中，实型变量分配存储空间字节数、表示数的范围等如表 2-3 所示。

表 2-3 实型变量类型

类型	存储空间字节数	表示数的范围	有效数字
float	4	$-10^{38}\sim10^{38}$	7 位
double	8	$-10^{308}\sim10^{308}$	15～16 位

我们已经介绍过实数可以用小数表示，也可以用指数表示，内存中实数一律以指数形式存储。

整数在计算机内存中可以精确存储，实数存储往往存在误差，看下面的例子：

[例 2.2] 输出实型变量（浮点型）数值。

```
#include "stdio.h"
main()
{   float a,b;
    a=1234.567e5;
    b=a+5;
    printf("%f\n",b);
}
```

程序运行情况如下：

```
123456712.000000
```

程序中 printf("%f\n",b);是按实型的格式输出 b 的值，结果是 123456712.000000，只有前 7 位准确，这是因为 float 类型只能保证 7 位有效数字。

3. 字符型变量

字符型变量用来存放字符常量，注意只存放单个字符，在内存中每个字符占一个字节，存放字符的 ASCII 码。

如前所述，对于字符型变量使用前也要先定义。

例如 char ch1,ch2;

它表示 ch1 和 ch2 为字符变量，各可以放一个字符，所以可以对 ch1、ch2 赋值：

```
ch1='A';
ch2='B';
```

在内存中存储形式如图 2-6 所示。

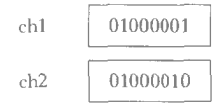

图 2-6 字符数据在内存中存储示意图

由图可见，字符数据在内存中是以二进制整数的形式存储，因此，字符型数据也可以像整型数据那样使用，它可以用来表示一些特定范围内的整数。

字符型变量分为两种：一般字符类型（char）和无符号字符类型（unsigned char），字符型变量的分类、字节长度和取值范围如表 2-4 所示。

表 2-4 字符型变量类型

数据类型	字节长度	取值范围
[signed] char	1	-128～127
[unsigned] char	1	0～255

C 语言中字符型数据和整型数据可以通用，一个字符型数据既可以以字符形式输出，也可以以整数形式输出，请看下面的例子：

[例 2.3] 字符型数据通用性示例

```
#include "stdio.h"
main()
{char c1,c2;
 c1 = 'A';
 c2 = 66;
 printf("%d,%c\n",c1,c1);
 printf("%d,%c\n",c2,c2);
}
```

程序运行情况如下：

```
65,A
66,B
```

c1、c2 被指定为字符变量，c1 = 'A'；这语句实际把 'A' 的 ASCII 码 65 送到 c1 中，c2 = 66 这句把 66 送到 c2 中，66 是 'B' 的 ASCII 码，printf("%d,%c\n",c1,c1);中%d 是十进制整数输出格式，%c 是字符输出格式。

[例 2.4] 把大写字母(A)转换为到小写字母（a）

```
#include "stdio.h"
main()
{char c1,c2;
 c1 = 'A';
 c2 = c1 + 32;
 printf("%c,%d\n",c1,c1);
 printf("%c,%d\n",c2,c2);
```

```
}
```

程序运行情况如下：

```
A,65
a,97
```

以上程序中运用小写字母比对应的大写字母大 32 的规律，进行大写字母到小写字母的转换。

三、数据类型的转换

整型、实型和字符型数据可以进行混和运算。在 C 语言中如果参加运算的两个量类型不一致，若一个是整型数，一个是实型数，系统自动把整型数转换为实型数，使运算量的类型达到一致，然后再进行运算，结果也是实型。即把较低的类型转换为较高的类型，然后再进行运算，结果是较高类型。不同类型进行混和运算遵循规则如图 2-7 所示。

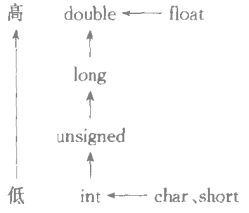

图 2-7 数据类型的转换规则

图中横向箭头表示必然要进行的转换，即 char 、short int 要转换为 int 型，float 要转换为 double 型数据，然后再参加运算。

纵向箭头表示参加运算的两个数据如果类型不同，要将低类型转换为高类型，例如参加运算的两个数一个是 int 类型，一个是 double 类型，则要将 int 类型的数据转换为 double 类型，然后按 double 类型进行运算。

按照这个规则，表达式 20 + 'b' - 1.5 + 10.87 * 'B'的运算次序为：

①进行 20 + 'b'运算，先将 'b' 转换为整数 98，结果为 118。

②进行 118 - 1.5 运算，将 118 转换为 double 类型，与 1.5 相减，结果 116.5。

③进行 10.87 * 'B'运算，'B' 转换为整数 66，再转换为 double 类型，再将 10.87 与之按 double 类型进行运算。

④将②、③两步的结果进行加法运算，结果为 double 类型，以上的类型转换由系统自动完成。

四、变量的初始化

程序中常需要对某些变量在定义时赋初值，即初始化，对变量初始化的一般形式如下：

数据类型　　变量 = 初值；

例如：

```
int a = 5;
float x = 4.56;
char c1 = 'A';
```

说明：

(1)几个变量同时赋予相同的初值，要分别赋初值。例如：

若写成：int a=b=c=5;是错误的，应该写成：int a=5,b=5,c=5;

(2)可以给定义的变量中的一部分赋初值，如：

int a,b,c=5;

(3)这里介绍的变量赋初值，是在程序运行时赋给的。（后面要介绍的静态存储变量和外部变量的赋初值是在编译阶段赋给的。）

五、局部变量、全局变量和存储类型

定义变量可以在函数内部或复合语句内部进行，也可以在函数外部进行。

我们把在函数内部或复合语句内部定义的变量称为局部变量，局部变量也称为内部变量，局部变量只在本函数或复合语句中有效，离开它所在函数或复合语句局部变量变为无效，释放其所占的内存单元。

在函数外部定义的变量称为全局变量。全局变量也称为外部变量。全局变量的有效范围是从定义变量的位置开始到本源程序文件结束。源程序文件是程序的编译单位。

当我们说明了一个变量的数据类型，C编译系统要给该变量分配若干字节的存储空间，用来存放该变量的值。计算机中的寄存器和内存单元都可以存放数据，而内存中用来放用户数据的数据区又分为静态存储区和动态存储区。在静态存储区中存放的变量在程序开始执行时分配存储单元，程序执行完毕时释放。在程序执行过程中它们占据固定的存储单元，而不是动态地分配和释放。在动态存储区中存放的变量在函数调用开始时分配动态存储空间，函数结束时释放这些空间。变量是存放在寄存器中，还是存放在内存中，是放在内存的静态存储区，还是动态存储区，在进行变量定义时也要定义。我们把变量存放在何处称为存储类型。加上对变量存储类型的定义，变量定义的一般形式如下：

存储类型标识符　数据类型　变量名［，变量名，……］;

数据类型前面已经介绍过了。存储类型标识符以后各章节中简称存储类型，C语言中存储类型符及含义如表 2-5 所示。

表 2-5　存储类型说明符

存储类型	存储类型符	存储区域
自动型	auto	动态存储区
寄存器型	register	CPU 的通用寄存器
静态型	static	静态存储区
外部参照型	extern	

1. auto 变量

局部变量可以说明为 auto 型，而全局变量不能被说明成此类型。

对于局部变量，如果没有指定存储类型、或使用了 auto 说明符，系统就认为所定义的变量具有自动类型。因此，