

第3章

顺序结构程序设计

教学目标

掌握 C 语言顺序结构程序设计的基本方法,数据输入与输出的方法及相应的常用库函数的使用。

本章要点

- C 语句概述
- 字符数据的输入与输出函数 `getchar()` 与 `putchar()` 的使用
- 格式输入与输出函数 `printf()` 与 `scanf()` 的使用

在进行程序设计时,有两部分工作,一部分是数据的设计,另一部分是操作的设计。数据设计是对一系列数据的描述,主要是定义数据的类型,完成数据的初始化等;操作的设计是产生一系列的操作控制语句,其作用是向计算机系统发出操作指令,以完成对数据的加工和流程控制。

当我们运用 C 语言编程解决实际问题时,必须组织相应的语句来完成数据的设计和操作的设计,这里的“组织相应的语句”便隐含了结构问题。一般地讲,C 程序可由顺序结构、选择结构和循环结构三种结构组成。学习要由浅入深,循序渐进;路要一步一步地走。当构成 C 程序的各语句间在运行时客观地存在一个先、后的次序,这些语句形成了顺序结构。顺序结构是最简单、最基本的程序结构,其包含的语句是按顺序执行的,且每条语句都将被执行。其他的结构可以包含顺序结构,也可以作为顺序结构的组成部分。本章主要讲述顺序结构程序设计方法。

3.1 C 语句分类概述

C 语句可以分为以下 5 类。

1. 控制语句

控制语句完成一定的控制功能。C 语言只有 9 种控制语句,分别如下。

- ① `if()~else~` (条件语句)

- ② switch (多分支选择语句)
- ③ goto (转向语句)
- ④ while()~ (循环语句)
- ⑤ do~while() (循环语句)
- ⑥ for()~ (循环语句)
- ⑦ break (中止执行 switch 或循环语句)
- ⑧ continue (结束本次循环语句)
- ⑨ return (从函数返回语句)

以上 9 种语句中的括号()表示其中是一个条件,~表示内嵌的语句。这些语句将在后面的章节中陆续介绍。

2. 变量声明语句

由类型关键字后接变量名(如果有多个变量名,则用逗号分隔)和分号构成的语句,如“int a,b,c;”。

3. 表达式语句

在任何一个 C 语言合法表达式的后面加一个分号就构成了相应的表达式语句。表达式语句的一般形式如下:

表达式;

注意: 一个语句必须在最后出现分号,分号是语句中不可缺少的一部分。例如:

```
i++;  
x+y;
```

在 C 语言中,最常用的表达式语句是赋值语句和函数调用语句。例如:

```
a=7;           /* 赋值语句 */  
putchar(a);    /* 函数调用语句 */
```

函数调用语句就是在函数调用的后面加一个分号。函数调用语句的一般形式如下:

函数名(参数列表);

C 语言有丰富的标准库函数,有用于键盘输入和显示器输出的库函数、求数学函数值的库函数、磁盘文件读写的库函数等,这些函数完成预先设定好的任务,用户可直接调用。需要注意的是,调用标准库函数时要在程序的开始处用#include 编译预处理命令将所调函数相应的头文件包含到程序中来。如:

```
#include "stdio.h"  
#include "math.h"
```

有了#include "stdio.h"在程序中才能调用标准输入输出函数;有了#include "math.h" 在程序中才能调用数学函数。关于头文件和标准库函数将在后续章节中再详细介绍。

要注意语句与表达式在概念上的区别,例如: a=b+5;是一条语句,而 a=b+5 是一个

赋值表达式。

4. 空语句

只有一个分号的语句是空语句,其一般形式如下:

```
;
```

空语句在语法上占有一个语句的位置,而执行该语句不做任何操作。空语句常用于循环语句中,构成空循环。

5. 复合语句

复合语句是由花括号{}将多条语句括在一起而构成的,在语法上相当于一条语句。复合语句的一般形式如下:

```
{ [内部数据描述]
  语句 1
  ...
  语句 n
}
```

注意:在复合语句的“内部数据描述”中定义的变量,仅在复合语句中有效;复合语句结束的“}”之后,不需要再加分号。

例 3.1 复合语句。

```
#include<stdio.h>
main()
{  int m=1;
    printf("m=%d\n",m);          /* 输出 m=1 */
    {  int m=2;
        printf("m=%d\n",m);      /* 输出 m=2 */
    }
    printf("m=%d\n",m);          /* 输出 m=1 */
}
```

运行结果:

```
m=1
m=2
m=1
```

在上面的程序中,主函数有 4 条语句,其中的第 3 条语句是复合语句。在复合语句中定义的变量 m 与主函数中定义的变量 m 对应的是不同的内存空间,因此互不影响。从程序的运行结果也可看出这一点。

复合语句常用于流程控制语句中执行多条语句的情况。

3.2 数据输入与输出

我们在前面已经介绍过，C 语言顺序结构程序设计中经常用到输入输出操作，下面将介绍与输入输出相关的一些函数。

所谓输入输出是以计算机主机为主体而言的。从计算机内部向计算机外部设备（如磁盘、打印机、显示屏等）输出数据的过程称为“输出”；从计算机外部设备（如磁盘、光盘、键盘、扫描仪等）向计算机内部输入数据的过程称为“输入”。

C 语言没有提供输入输出语句，数据的输入和输出是通过调用输入输出函数实现的，即在输入输出函数的后面加上“;”，这些函数包含在标准输入输出库中。这样处理，一方面可以使 C 语言的内核比较精练，另一方面也为 C 语言程序的可移植性打下了基础。

在 C 语言标准函数库中提供了一些输入输出函数，如 printf 函数和 scanf 函数，它们不是 C 语言的关键字，而只是函数的名字。实际上完全可以不用 printf 和 scanf 这两个名字，而另外编两个输入输出函数，用其他的函数名。

在 VC 6.0 环境下，如果要在程序中使用输入输出函数，应首先用编译预处理命令 #include 将头文件 stdio.h 包含到源文件中，因为在该头文件中包含了与输入输出函数有关的信息。因此，在调用标准输入输出库函数时，应在文件开头包含以下预处理命令：

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

stdio.h 是 standard input&output 的缩写，它包含了与标准 I/O 库有关的变量定义和宏定义。

常用的输入输出函数有：scanf（格式输入）、printf（格式输出）、getchar（字符输入）、putchar（字符输出）、gets（字符串输入）、puts（字符串输出）。本章主要介绍前 4 个最基本的输入输出函数。

3.2.1 字符输出函数 putchar()

函数原型：int putchar(int);。

函数功能：向标准输出设备（一般为显示器）输出一个字符，并返回输出字符的 ASCII 码值。

函数的参数可以是字符常量、字符变量或整型变量，即将一个整型数作为 ASCII 码值，输出相应的字符。例如：

```
#include "stdio.h"
main()
{   char ch='B';
    int i=67;
    putchar(i);           /* 输出 ASCII 码值为 67 所对应的字符 C */
    putchar('\n');        /* 输出控制字符,起换行作用 */
    putchar(ch);          /* 输出字符变量 ch 的值 B */
    putchar('\x42');      /* 输出字母 B */
}
```

```
    putchar(0x42);           /* 直接用 ASCII 码值输出字母 B */
}
```

运行结果:

```
A
BBB
```

3.2.2 字符输入函数 getchar()

函数原型: `int getchar(void);`

函数功能: 从标准输入设备(一般为键盘)输入一个字符,函数的返回值是该字符的 ASCII 码值。这里的 `void` 是类型名,表示空类型,即该函数不需要参数。

字符输入函数每调用一次,就从标准输入设备上取一个字符。函数值可以赋给一个字符变量,也可以赋给一个整型变量。例如:

```
#include "stdio.h"
main()
{   char i;
    int r;
    i=getchar();           /* 将键盘输入字符的 ASCII 码值赋给 ch */
    r=getchar();           /* 将键盘输入字符的 ASCII 码值赋给 i */
    putchar(i);
    putchar('\n');
    putchar(r);
}
```

运行该程序时,若输入如下:

```
ab ↵
```

则变量 `i` 的值为 97,变量 `r` 的值为 98,输出的结果如下:

```
a
b
```

注意: 用 `getchar()` 输入字符结束后需要按 Enter 键,程序才会响应输入,继续执行后续语句。

字符输入和字符输出函数使用非常方便,但每一次函数调用只能输入或输出一个字符。

3.2.3 格式输出函数 printf()

1. 格式输出函数的一般形式

函数原型: `int printf(char * format[, argument, ...]);`。

函数功能: 按规定格式向输出设备(一般为显示器)输出数据,并返回实际输出的字符数;若出错,则返回负数。

printf()函数使用的一般形式如下:

```
printf("格式控制字符串",输出项表列);
```

例如:

```
int i=97;
printf("i=%d,%c\n", i, i);
```

|
|
|

格式控制字符串
输出项表列

调用函数 printf("i=%d,%c\n",i,i); 中的两个输出项都是变量 i,但却以不同的格式输出,%d 控制的 i 输出的是整型数 97,%c 控制的 i 输出的是字符 a,格式控制字符串中的“i=”是普通字符,它将按原样输出,\n 是转义字符,它的作用是换行。

2. 格式控制字符串

“格式控制字符串”必须用英文状态下的双引号括起来,它的作用是控制输出项的格式和输出一些提示信息。它一般由三部分组成:转义字符、格式说明、普通字符。

(1) 转义字符。转义字符是以“\”开始的字符,用来指明特定的操作,如\n表示换行,\t表示水平制表等。转义字符见表 2-2。

(2) 格式说明。由“%”和格式字符组成,用来指定数据的输出格式。

在 C 语言中,应根据输出数据类型的不同选用不同的格式字符来控制输出格式。表 3-1 列出了在 printf()函数中可以使用的格式字符。

表 3-1 printf 格式字符

格式字符	说 明
d,i	以带符号的十进制形式输出整数(正数不输出符号)
o	以无符号八进制形式输出整数(不输出前导符 0)
x,X	以无符号十六进制形式输出整数(不输出前导符 0x),用 x 则输出十六进制数的 a~f 时以小写形式输出。用 X 时,则以大写形式输出
u	以无符号的十进制形式输出整数
c	输出一个字符
s	输出字符串
f	以十进制小数形式输出单、双数,隐含输出 6 位小数,输出的数字并非全部是有效数字,单精度实数的有效位数一般为 7 位,双精度实数的有效位数一般为 16 位
e, E	以指数形式输出单、双精度数。如用“e”,输出时指数以小写“e”表示;如用“E”,输出时指数以大写“E”表示
g, G	选用%f%e 格式中输出宽度较短的一种格式,不输出无意义的 0。用 G 时,若以指数形式输出,则指数以大写表示
%	输出百分号(%)

例如:

```
printf("%d%o%x",a,b,c);
```

这表示输出项表列 a、b、c 分别按十进制(%d)、八进制(%o)和十六进制(%x)输出。

在格式控制字符串中的%和格式字符之间还可以插入以下几种附加字符(又称修饰符)。见表 3-2。

表 3-2 printf 的附加格式说明字符

字 符	说 明
字母 l	表示长整型数据,可加在格式字符 d、o、u、x 前面
m(代表一个正整数)	指定输出数据的最小宽度
n(代表一个正整数)	对实数,表示输出 n 位小数;对字符串,表示截取的字符个数
—	输出的数字或字符在域内向左靠

(3) 普通字符。普通字符在输出时,按原样输出,主要用于输出提示信息。

例如:

```
i=2;
printf("i=%d",i);
```

在格式控制字符串中,“i=”是普通字符按原样输出;“%d”指定 i 按十进制数的形式输出,即输出 2;该语句的执行结果是“i=2”。

3. 输出项表列

“输出项表列”列出了要输出的各项数据,这些数据可以是常量、变量、表达式、函数返回值等。输出项可以是 0 个、一个或多个,每个输出项之间用逗号“,”分隔。输出的数据可以是整数、实数、字符和字符串。

4. 使用 printf 函数应注意的问题

(1) 在格式控制字符串中,格式说明和输出项在类型上必须一一对应。

(2) 在格式控制字符串中,格式说明的个数和输出项的个数应该相同。如果不同,则系统作如下处理。

- ① 如果格式说明的个数少于输出项数,多余的数据项不输出。
- ② 如果格式说明的个数多于输出项数,对多余的格式将输出不定值或 0 值。
- (3) 为整数指定输出宽度。

在%和格式符之间插入一个整数用来指定输出的宽度,如果指定的宽度多于数据实际宽度,则输出的数据右对齐,左端用空格补足;而当指定的宽度不足时,则按实际数据位数输出,这时指定的宽度不起作用。

例如:

```
#include<stdio.h>
main()
{ int a,b;
  a=123;
  b=12345;
```

```
printf("%4d,%4d",a,b);
}
```

程序中变量 a 按 4 位输出,由于其值为 3 位,因此左边补一个空格。变量 b 本身是 5 位,按指定宽度 4 位输出时宽度不够,因此按实际位数输出。所以执行结果如下:

```
_123,12345
```

其中“_”表示空格,下同。

(4) 为实数指定输出宽度。

对于 float 或 double 型数据,在指定数据输出宽度的同时,也可以指定小数位的位数,指定形式如下:

```
%m.nf
```

表示数据输出总的宽度为 m 位,其中小数部分占 n 位。当数据的小数位多于指定宽度 n 时,截去右边多余的小数,并对截去的第一位小数做四舍五入处理;而当数据的小数位少于指定宽度时,在小数的右边补零。

例如:

```
printf("%5.3f\n",12345.6789);
```

此语句的格式说明为 %5.3f,表示输出总宽度为 5,小数位为 3,这样整数部分只有 1 位,小于实际数据位数,只能按实际位数输出,而小数部分指定输出 3 位,将小数点后的第 4 位四舍五入,所以结果为“12345.679”。

(5) 输出对齐方式。

通过格式字符指定了输出宽度后,如果指定的宽度多于数据的实际宽度,则在输出时数据自动右对齐,左边用空格补足,此时,也可以指定将输出结果左对齐,方法是在宽度前加上“-”符号。

下面就常见的“格式控制字符串”的使用举几个例子。

例 3.2 整型数据的输出。

```
#include<stdio.h>
main()
{ int a=12;
  long b=20040978;
  printf("a=%d,a=%6d,a=%-6d,a=%06d\n",a,a,a,a);
  printf("%d,%o,%x,%u\n",a,a,a,a);
  printf("b=%ld\n",b);
}
```

运行结果如下:

```
a=12,a=    12,a=12    ,a=000012
12,14,c,12
b=20040978
```

例 3.3 实型数据的输出。

```
#include<stdio.h>
main()
{   float x=1234.567;
    double y=1234.5678;
    printf("%f,%f\n",x,y);
    printf("%6.3f,%10.3f\n",x,y);          /* 指定的宽度不足时,按实际数据输出 */
    printf("%e\n",x);
}
```

运行结果如下:

```
1234.567000,1234.567800
1234.567,  1234.568
1.23457e+03
```

例 3.4 字符数据的输出。

```
#include<stdio.h>
main()
{   char c='B';
    int i=65;
    printf("%c,%d\n",c,c);
    printf("%d,%c\n",i,i);
    printf("%-5c,%5c\n",c,c);
}
```

运行结果如下:

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

例 3.5 字符串的输出。

```
#include<stdio.h>
main()
{   printf("computer\n");
    printf("%s\n","computer");
    printf("%5s\n","computer");
    /* 指定的宽度小于字符串的长度,按原样输出 */
    printf("%10s\n","computer");
    /* 输出字符串时,占 10 个字符宽,右对齐,不足部分左端用空格占位 */
    printf("%-10s\n","computer");
    /* 输出字符串时,占 10 个字符宽,左对齐,不足部分右端用空格占位 */
    printf("%-10.5s\n","computer");
    /* 占 10 个字符宽且只输出前 5 个字符,左对齐,不足部分右端用空格占位 */
}
```

运行结果如下:

```
computer
computer
computer
  computer
computer
compu
```

3.2.4 格式输入函数 scanf()

1. 格式输入函数的一般形式

函数原型: `int scanf(char *format[, argument, ...]);`

函数功能: 按规定格式从键盘输入若干任何类型的数据给 argument 所指的单元。返回输入并赋给 argument 的数据个数; 遇文件结束返回 EOF; 出错返回 0。

scanf() 函数使用的一般形式如下:

```
scanf("格式控制字符串", 地址表列);
```

例如:

```
scanf("%d,%d", &a, &b);
```

```
      |           |
      |           |
格式控制字符串  地址表列
```

假设从键盘输入“12, 65”, 系统会将 12 和 65 以 %d(十进制)形式读入, 并赋予变量 a 和 b 所代表的存储空间中。

2. 格式控制字符串

格式控制字符串的作用与 printf() 函数中的作用相似, 它一般由普通字符、输入数据格式说明组成。

(1) 格式说明。

输入数据格式说明以 % 开始, 以一个格式字符结束, 中间可以插入附加的字符。这里格式说明的作用是控制输入数据的格式。scanf() 函数中可以使用的格式字符及附加字符(修饰符)如表 3-3 和表 3-4 所示。

表 3-3 scanf 格式字符

格式字符	说 明
d, i	用来输入有符号的十进制整数
u	用来输入无符号的十进制整数
o	用来输入无符号的八进制整数
x, X	用来输入无符号的十六进制整数(大小写作用相同)
c	用来输入单个字符

格式字符	说 明
s	用来输入一个字符串。在输入时以非空白字符开始,以第一个空白字符结束。字符串以串结束标志'\0'作为其最后一个字符
f	用来输入实数,可以用小数形式或指数形式输入
e, E,g, G	与 f 作用相同,可以互相替换(大小写作用相同)

表 3-4 scanf 的附加格式说明字符

字 符	说 明
字母 l	用于输入长整型数据(%ld,%lo,%lx,%lu)以及 double 型数据(%lf 或 %le)
字母 h	用于输入短整型数据(%hd,%ho,%hx)
m(代表一个正整数)	指定输入数据所占宽度(列数)
*	表示本输入项在读入后不赋给任何变量

(2) 普通字符。

scanf()格式控制字符串中的普通字符是规定了输入时必须输入的字符。例如：

```
scanf("a=%d",&a);      /* “&”是取地址运算符,作用是得到 a 变量的内存地址 */
```

执行该语句时,若要将 30 输入到 a 变量中,应按下列格式输入：

```
a=30↵
```

这里↵表示按 Enter 键的意思。

若有语句

```
scanf("%d,%f",&a,&x);
```

要将 10 赋给 a,2.5 赋给 x,则对应的输入格式如下：

```
10,2.5↵
```

3. 地址表列

地址表列是由若干地址组成的列表,可以是变量的地址、字符串的首地址、指针变量等,各地址间用逗号隔开。

格式输入函数 scanf()是将键盘输入的数据流按格式转换成数据,存入与格式相对应的地址指向的存储单元中。所以下列 scanf()函数的调用是错误的：

```
scanf("%d,%d",a,b);
```

a,b 表示的是变量 a 和 b 的值,不是地址。正确的用法是：

```
scanf("%d,%d",&a,&b);
```

初学者要注意用 scanf()函数和 printf()函数进行数据输入、输出时的不同之处。

```
scanf("%d",&a);          /* 从键盘输入数据,存入 a 变量的内存地址中 */  
printf("%d",a);         /* 将变量 a 的值输出 */
```

4. 使用 scanf() 函数应注意的问题

(1) 输入多个数据时的分隔处理。

若用一个 scanf() 函数输入多个数据,且格式说明之间没有任何普通字符,则输入时,数据之间需要用分隔符,例如语句:

```
scanf("%d%d",&a,&b);
```

执行该语句时,输入的两个数据之间可用一个或多个空格分隔,也可以用 Enter 键分隔,例如:

```
5 10↵
```

或

```
5↵  
10↵
```

当 scanf() 函数指定输入数据所占的宽度时,将自动按指定宽度来截取数据。例如:

```
scanf("%2d%3d",&a,&b);
```

若输入为

```
123456789↵
```

则函数截取 12 存入地址 &a 中,截取 345 存入地址 &b 中。

(2) 输入实型数时不能规定精度。

用 scanf() 函数输入实型数时,可以指定宽度,但不能规定精度。例如:

```
scanf("%4f%5f",&x,&y);
```

是正确的,若输入

```
12.345.6789↵
```

则 12.3 赋给变量 x,45.67 赋给变量 y。而语句

```
scanf("%10.2f",&x);
```

是错误的。

(3) 用“%c”格式输入字符。

在用“%c”格式输入字符时,空格字符和转义字符都作为有效字符输入。例如:

```
scanf("%c%c%c",&a,&b,&c);
```

若输入

```
a b c↵
```

则字符'a'赋给 a,空格赋给 b,字符'b'赋给 c,而后面的'c'已无意义。这是因为%c 只要求读入一个字符,后面不需要用空格作为两个字符间的间隔。若将输入改为

abc↵

则字符'a'赋给 a,字符'b'赋给 b,字符'c'赋给 c。

“%c”格式与其他格式混合使用时,也存在类似问题。例如:

```
int a,b;  
char ch;  
scanf("%d%c%d",&a,&ch,&b);
```

若要将 12 存入地址 &a 中,a 存入地址 &ch 中,34 存入地址 &b 中,应按如下方法输入:

12a34↵

(4) 附加字符“*”的用法。

格式说明中的附加字符“*”为输入赋值抑制字符,表示该格式说明要求输入数据,但不赋值。例如:

```
scanf("%3d%*2d%f",&a,&x);
```

若输入为

12345678.9↵

则 123 赋给 a,678.9 赋给 x,而 45 不赋给任何变量。

(5) 从键盘输入数据的个数应该与函数要求的个数相同,当个数不同时系统做如下的处理:

- 如果输入数据少于 scanf 函数要求的个数时,函数将等待输入,直到满足要求或遇到非法字符为止。
- 如果输入数据多于 scanf 函数要求的个数时,多余的数据将留在缓冲区作为下一次输入操作的输入数据。

(6) 在输入数据时,遇到以下情况时该数据认为结束。

- 遇到空格、或按 Enter 键或按 Tab 键。
- 按指定的宽度结束,如“%3d”,只取 3 列。
- 遇到非法输入。

从本节中看出,C 语言的格式输入输出的规定比较烦琐,而输入输出又是程序中最基本的操作,因此关于格式输入输出在本节中做了比较详细的介绍。要想在程序中很好地运用输入输出函数,还应通过上机编写程序和调试程序来逐步深入而自然地掌握这些函数的用法。

3.3 程序举例

有了以上介绍的输入输出函数,我们已可以用表达式语句和函数调用语句等来编写顺序结构程序。

一个顺序结构程序,一般包括以下两部分。

1. 编译预处理命令

在程序的编写过程中,若要使用标准库函数,需要用编译预处理命令 `#include`,将相应的头文件包含进来。若程序中只使用 `scanf()` 函数和 `printf()` 函数,可省略不写 `#include "stdio.h"`。

2. 主函数

在主函数体中,包含着顺序执行的各条语句。主要有以下几部分。

- (1) 变量类型说明。
- (2) 给变量提供数据。
- (3) 按题目要求进行运算。
- (4) 输出运算结果。

例 3.6 从键盘输入三角形的三边长,求三角形周长(假设输入的三边长能构成三角形)。

算法分析:

- (1) 定义实型变量 `a, b, c, perimeter`, 用于存放输入的数据和计算的结果;
- (2) 输入三个实型数存入变量 `a, b, c` 中, 要求满足 $a+b>c, b+c>a, c+a>b$;
- (3) 求周长 $a+b+c$ 存入变量 `perimeter` 中;
- (4) 输出 `perimeter`。

参考程序:

```
#include<stdio.h>
#include "math.h"
main()
{ float a,b,c, perimeter;
  printf("Input a,b,c:");
  scanf("%f,%f,%f",&a,&b,&c);          /* 输入三角形的三边长 */
  perimeter =a+b+c;                    /* 求三角形周长 perimeter */
  printf("perimeter =%.2f\n", perimeter);
}
```

例 3.7 从键盘上输入一个大写字母,输出对应的小写字母。

算法分析:

- (1) 定义字符型变量 `c`。
- (2) 输入大写字母存入变量 `c`。

(3) 转换成小写 $c=c+32$;

(4) 输出变量 c 。

参考程序:

```
#include "stdio.h"
main()
{   char c;
    printf("Input a uppercase letter: ");
    c=getchar();           /* 也可用 scanf("%c",&c);语句 */
    c=c+32;                /* 将大写字母转换成对应的小写字母 */
    putchar(c);            /* 也可用 printf("%c",c);语句 */
}
```

例 3.8 求方程 $ax^2+bx+c=0$ 的实数根。 a,b,c 的值由键盘输入。

算法分析:

- (1) 定义实型变量 $a,b,c, disc, x1, x2$, 用于存放输入的数据和计算的结果;
- (2) 输入三个实型数存入变量 a,b,c 中, 要求满足 $a \neq 0$ 且 $b^2-4ac > 0$;
- (3) 求判别式 b^2-4ac 的值存入变量 $disc$ 中;
- (4) 调用求平方根函数 $\text{sqrt}()$, 求方程的根存入变量 $x1, x2$ 中;
- (5) 输出 $x1, x2$ 。

参考程序:

```
#include<stdio.h>
#include "math.h"
main()
{   float a,b,c,disc,x1,x2;
    printf("Input a,b,c:");
    scanf("%f,%f,%f",&a,&b,&c);    /* 输入方程的三个系数 */
    disc=b*b-4*a*c;               /* 求判别式的值 */
    x1=(-b+sqrt(disc))/(2*a);      /* 求方程的第一个实数根 */
    x2=(-b-sqrt(disc))/(2*a);      /* 求方程的第二个实数根 */
    printf("\nx1=%6.2f\nx2=%6.2f\n",x1,x2);
}
```

由于程序中使用了数学函数 $\text{sqrt}()$, 因此在程序的开头加了一条 $\#include$ 命令, 将头文件 math.h 包含到程序中来。

本章小结

顺序结构是最简单、最基本的程序结构, 赋值操作和输入/输出操作是顺序结构中最典型的结构。

1. C 语句可以分为以下 5 类

- (1) 控制语句 9 种。

- (2) 变量声明语句。
- (3) 表达式语句。
- (4) 空语句。
- (5) 复合语句。

2. 字符输入输出函数

- (1) 字符输出函数 putchar()。
- (2) 字符输入函数 getchar()。
- (3) 格式输出函数 printf()。

printf()函数使用的一般形式如下:

```
printf("格式控制字符串",输出项表列);
```

- (4) 格式输入函数 scanf()。

scanf()函数使用的一般形式如下:

```
scanf("格式控制字符串",地址表列);
```

3. 顺序结构程序一般包括两部分

- (1) 编译预处理命令。

在程序的编写过程中,若要使用标准库函数,需要用编译预处理命令 #include,将相应的头文件包含进来。

- (2) 主函数。

在主函数体中,包含着顺序执行的各条语句。主要有以下几部分。

- ① 变量类型说明。
- ② 给变量提供数据。
- ③ 按题目要求进行运算。
- ④ 输出运算结果。

习题

一、选择题

- (1) C语言规定:在一个源程序中,main 函数的位置()。
 - A. 必须在系统调用的库函数的后面
 - B. 必须在最开始
 - C. 可以任意
 - D. 必须在最后
- (2) 以下程序运行的结果是()。

```
#include <stdio.h>
int main()
{   int a=201,b=012;
    printf("%2d,%2d\n",a,b);
```

```
    return 0;  
}
```

- A. 01,12 B. 201,10 C. 01,10 D. 20,01

(3) 根据定义和数据的输入方式,输入语句的正确形式为()。

已有定义:

```
float f1, f2;
```

数据的输入方式:

4.52<回车>

3.5<回车>

- A. scanf("%f,%f",&f1,&f2);
B. scanf("%f%f",&f1,&f2);
C. scanf("%3.2f%2.1f",&f1,&f2);
D. scanf("%3.2f,%2.1f",&f1,&f2);

(4) 已知 int a,b;用语句 scanf("%d%d",&a,&b);输入 a,b 的值,不能作为输入数据分隔符的是()。

- A. , B. 空格 C. Enter 键 D. "[tab]"

(5) 以下叙述中错误的是()。

- A. C 语句必须以分号结束
B. 复合语句在语法上被看作一条语句
C. 空语句出现在任何位置都不会影响程序运行
D. 赋值表达式末尾加分号就构成赋值语句

(6) 以下叙述中正确的是()。

- A. 调用 printf 函数时,必须要有输出项
B. 调用 putchar 函数时,必须在之前包含头文件 stdio.h
C. 在 C 语言中,整数可以以十二进制、八进制或十六进制的形式输出
D. 调用 getchar 函数读入字符时,可以从键盘上输入字符所对应的 ASCII 码值

(7) 有以下程序:

```
#include<stdio.h>  
main()  
{  
    int m,n,p;  
    scanf("m=%dn=%dp=%d",&m,&n,&p);  
    printf("%d%d%d\n",m,n,p);  
}
```

若想从键盘上输入数据,使变量 m 中的值为 123,n 中的值为 456,p 中的值为 789,则正确的输入是()。

- A. m=123n=456p=789 B. m=123 n=456 p=789
C. m=123,n=456,P=789 D. 123 456 789

(8) 有以下程序:

```
#include<stdio.h>
main()
{
    char a,b,c,d;
    scanf("%c,%c,%d,%d",&a,&b,&c,&d);
    printf("%c,%c,%c,%c\n",a,b,c,d);
}
```

若运行时从键盘输入 6,5,65,66<Enter 键>,则输出结果是()。

- A. 6,5,A,B B. 6,5,65,66 C. 6,5,6,5 D. 6,5,6,6

二、填空题

(1) 在 C 语言中,调用输入/输出库函数,应在程序的开始处有预编译命令_____。

(2) 以下程序运行后的输出结果是_____。

```
int x=100,y=200;
printf("%d", (x,y));
```

(3) 有以下程序:

```
#include<stdio.h>
main()
{
    char ch1,ch2;
    int n1,n2;
    ch1=getchar();
    ch2=getchar();
    n1=ch1-'0';
    n2=n1*10+(ch2-'0');
    printf("%d\n",n2);
}
```

程序运行时输入 12<Enter 键>,输出结果是_____。

三、编程题

(1) 已知一个直角三角形的两个直角边分别为 $a=5$, $b=7$,求直角三角形的面积 s 。

(2) 输入两个整数,输出它们的平均数,保留一位小数。

(3) 输入一个字母,输出它的后继字母。如输入 a ,则输出 b 。

(4) 已知一个圆柱体的半径 $r=10$,高 $h=15$,求圆柱体的底周长 c ,底面积 s ,侧面积 s_1 ,表面积 s_2 ,体积 v 。

(5) 编程输入一个数字字符(0~9),将其转换为相应的整数后显示出来。

(6) 输入一个华氏温度,要求输出摄氏温度。公式如下:

$$c=5/9\times(f-32)$$

输出要有文字说明,取两位小数。

(7) 编写程序,实现从键盘输入学生的三门课成绩,计算输出其总成绩 sum 和平均成绩 ave。

(8) 求前驱字符和后继字符。输入一个字符,找出它的前驱字符和后继字符,并按 ASCII 码值从大到小的顺序输出这三个字符及其对应的 ASCII 码值。