第 3 章　简单程序设计

- 掌握 C 语言中的语句类型、程序结构
- 掌握赋值语句和基本输入/输出函数的使用
- 学会用正确的格式进行简单的输入输出程序设计

3.1　C 语言语句

语句是完成一定任务的命令。语句书写的特点是以分号（;）作为结束符。

C 语言的语句可分为 5 种类型，下面详细介绍。

1．表达式语句

由表达式组成的语句称为表达式语句，其作用是计算表达式的值或改变变量的值。它的一般形式是：

表达式;

注意没有分号不能称为语句。例如：

```
x=100          /*表达式*/
x=100;         /*语句*/
```

2．函数调用语句

由一个函数调用加上一个分号构成函数调用语句，其作用是完成特定的功能。它的一般形式是：

函数名(参数列表);

例如：

```
printf("Hello World!\n");    /*调用库函数，输出字符串*/
```

3．控制语句

控制语句用于完成一定的控制功能，以实现程序的结构化。

C 语言有 9 种控制语句，可分为以下 3 类：

（1）条件判断语句：**if** 语句、**switch** 语句。

（2）转向语句：**break** 语句、**continue** 语句、**goto** 语句、**return** 语句。

（3）循环语句：**for** 语句、**while** 语句、**do-while** 语句。

4．复合语句

复合语句是用花括号将若干语句组合在一起，又称分程序，形式上是几条语句，但在语法上可相当于一条语句。例如，下面是一个复合语句：

```
{
    i=5;
```

```
    printf("%d\n",i);
}
```

在后面选择结构和循环结构的学习中要特别注意该类语句。

5．空语句

只有一个分号的语句称为空语句。它的一般形式是：

```
;
```

例如：

```
x=100; ;            /*两条语句，后面是一条空语句*/
```

空语句是不执行任何命令的语句，常用于占位、循环语句中的循环体等。例如：

```
while (getchar() != '\n')
    ;               /*空语句*/
```

该循环的功能是：当从键盘上输入回车符后退出循环，否则循环一直继续。该空语句是对整个结构语法上的完善，是必不可少的一部分。上面的程序如果写成：

```
while (getchar() != '\n')
    printf("Hello World!\n");
```

程序将不断输出"Hello World!\n"，直到输入回车符。但如果写成：

```
while (getchar() != '\n')
    ;
printf("Hello World!\n");
```

则变成：当输入回车符后结束循环，否则什么事也不做；当结束循环后，只输出一行"Hello World!\n"。

3.2 程序结构

3.2.1 程序结构简介

在 C 语言中，程序结构一般分为顺序结构、选择结构、循环结构。任何复杂的程序都是由这三种基本结构组成的。

【例 3-1】简单的程序结构。

```
#include <stdio.h>
void main()
{
    int a,b,c;      /*声明部分，定义了 3 个整型变量*/
    a=100;          /*执行部分开始，直到最后的花括号*/
    b=200;
    c=a+b;
    printf("a+b=%d\n",c);
}
```

该程序的作用是求两个整数 a 与 b 的和 c。程序只有一个主函数 main，函数分成声明部分和执行部分。声明部分定义了变量 a、b、c 是 int 类型变量。执行部分包括 3 个赋值语句，使 a、b 的值分别为 100 和 200 并使 c 的值等于 a+b。最后一行输出变量 c 的值。程序运行后的结果是：

```
a+b=300
```

【例 3-2】由多个函数构成的程序结构。

```
#include <stdio.h>
void main()                    /*主函数*/
{
    int a,b,c;                 /*声明部分，定义变量的类型*/
    scanf("%d,%d",&a,&b);      /*通过输入函数，给变量 a、b 赋值*/
    c=sum(a,b);                /*调用 sum 函数，将函数值赋给变量 c*/
    printf("a+b=%d\n",c);      /*输出变量 c 的值*/
}
int sum(int a,int b)           /*定义一个 sum 函数*/
{
    int c;
    c=a+b;
    return (c);                /*将变量 c 的值通过返回语句带回调用处*/
}
```

本程序包含两个函数：主函数 main 和被调用函数 sum。

sum 函数的作用是将 a 和 b 的和赋值给变量 c，并通过返回语句 return 将 c 的值返回给主函数 main。

程序运行时，先由 scanf 函数从键盘上读取两个整型数据，如从键盘上输入：

```
100,200 <回车>
```

此时 a 被赋值 100，b 被赋值 200，然后执行语句 c=sum(a,b);，对 sum 函数进行调用，调用的结果是将和 300 赋给变量 c。

程序输出的结果是：

```
a+b=300
```

结果同前面的例子，只是程序的设计方法不同。

从上面的两个例子看出：一个 C 程序可以由若干个源程序文件组成，其结构如图 3-1 所示。

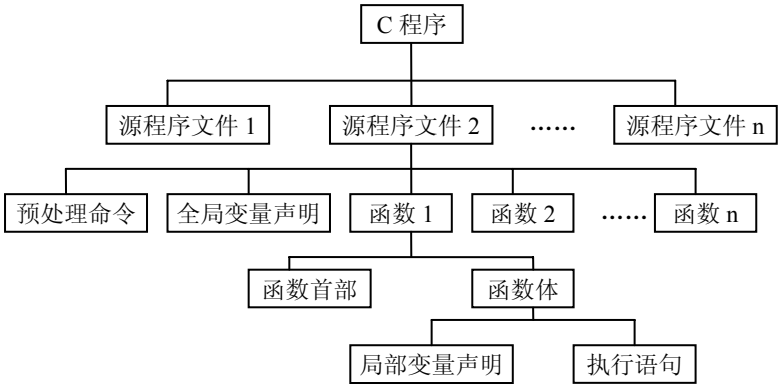

图 3-1　C 语言程序结构

3.2.2　顺序结构

顺序结构是程序设计中最简单、最基本的结构，其特点是程序运行时按语句书写的次序

依次执行，其结构如图 3-2 所示。在图 3-2 中，执行完 A，按序执行 B。顺序结构通常是由简单语句、复合语句及输入输出函数语句组成。

【例 3-3】分析下面程序结构。

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

上述程序中主函数内的几条语句是顺序结构，其语句执行的次序如图 3-3 所示。

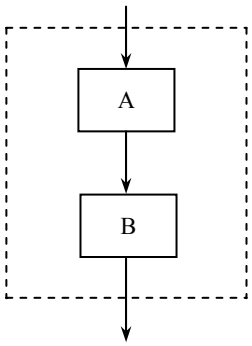

图 3-2　顺序结构流程图

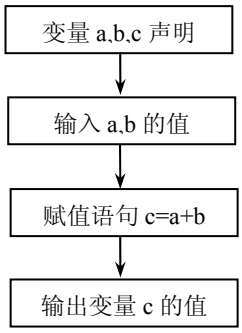

图 3-3　例 3-3 的流程

注意：#include <stdio.h> 称为预处理命令，而不是语句。

3.3　赋值语句

3.3.1　基本赋值语句

赋值语句是程序设计中最常用的语句。其一般形式为：

变量 = 表达式;

赋值语句的功能是将赋值号右边表达式的值计算出来，再赋给赋值号左边的变量。如：

```
c=a+b;
```

该语句的作用是将表达式 a+b 计算后的结果赋给变量 c。

以下是正确的赋值语句：

```
a=100;a=a+200;              /*两个赋值语句，最后 a 变成 300*/
a=b=c=100;                  /*相当于 a=(b=c=100); */
c=(a=100,b=a,a+b);          /*右边是逗号表达式，表达式的值是 a+b*/
```

下面是错误的赋值语句：

```
c+2=a+b;                    /*左边不是变量名，是表达式*/
```

要注意：

```
a=b=c=100
a=b=c=100;
```

前者是赋值表达式，相当于 a=(b=(c=100))，最右边的=先运算。后者是赋值语句，相当于 a=(b=c=100);，“b=c=100”是一个赋值表达式，整条语句相当于把一个赋值表达式的值赋给变量 a。

我们可以写成：

```
d=(a=b=c=100);
```

但不能写成：

```
d=(a=b=c=100;);
```

注意：语句是一条命令，是程序执行的最小单位，不能再被其他命令直接引用。其实赋值运算符“=”可以理解成←，例如：c=a+b 可以看成 c←a+b。后面我们还会遇到“==”运算符，其功能才相当于数学中的“=”号，例如：

```
c+2 == a+b
```

3.3.2 复合赋值语句

除了基本赋值语句之外，还可以用复合赋值运算符构造复合赋值语句。例如：

```
a+=3;          /*相当于 a=a+3 */
b-=6;          /*相当于 b=b-6 */
c/=2;          /*相当于 c=c/2 */
```

在构造以上赋值语句之前，变量必须已经初始化或赋值。下面的程序是错误的：

```
int a;
a+=10;
```

因为 a+=10 相当于 a=a+10，而右边表达式中的 a 是刚刚定义的，还没有具体的值。

3.4 数据的输入与输出

为了实现人机交互，程序设计中经常需要通过输入输出语句来实现数据的输入和输出。

数据输入。指从输入设备（例如键盘、磁盘、光盘、扫描仪等）向计算机输送数据。

数据输出。指从计算机向外部输出设备（例如显示器、打印机、磁盘等）输送数据。

高级程序设计语言的数据输入输出都是通过输入输出语句来实现的，而 C 语言本身不提供输入输出语句，其数据的输入和输出功能是由函数来实现的，这使得 C 语言编译系统简单、可移植性好。

C 语言提供的函数以库的形式存放在系统中，它们不是 C 语言文本中的组成部分。在使用函数库时，要用预编译命令#include 将有关的“头文件”包含到用户源文件中，例如：

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

预编译命令一般放在程序的开头，使用不同类型的函数需要包含不同的“头文件”。例如：

使用标准输入输出库函数 printf（格式输出）、scanf（格式输入）、putchar（输出字符），getchar（输入字符）等时，要用到 stdio.h 文件（考虑到 printf、scanf 使用频繁，Turbo C 中允许在使用这两个函数时不用#include 命令，而 Visual C++中则不能省略）。

使用数学函数库时，要用到 math.h 文件。

文件后缀中“h”是 head 的缩写，读者可以参考查阅附录中的函数列表。

3.4.1 格式化输出函数 printf

printf 函数的功能是向系统指定的设备输出若干个任意类型的数据。

printf 后面的字母 f 表示“**format**”，是“格式”的意思。

1．printf 函数调用形式

printf 函数是一个标准库函数，其调用的一般形式为：

printf(格式控制字符串,输出列表);

括号里格式控制字符串和输出列表实际上都是函数的参数。其中：

（1）格式控制字符串是用双引号括起来的字符串，它包括两个信息：

① 格式说明。由“%”和格式字符组成，如%d、%c、%f 等。它的作用是将要输出的数据转化成指定的格式输出，格式说明都是由“%”字符开始的。

② 一般字符。或者称为非格式说明符，即按原样输出的字符。

（2）输出列表是需要输出的变量、函数、表达式。

例如：

```
printf("a+b=%d\n",c);
```

①“%d”是格式说明，用来控制输出项 c 的输出格式。

②“a+b=”和“\n”都是一般字符，原样输出，“\n”是转义字符，代表换行符。

假设 c 为 300，则输出结果为：

```
a+b=300
```

2．格式说明

不同类型的数据用不同的格式说明。格式说明是由“%”开头，后面跟若干个英文字母，用以说明数据输出的类型、长度、位数等。其一般形式为：

% [标志] [最小宽度] [.精度] [长度] 类型

【说明】

[ ]：表示可选项。

[标志]：可以是-、+、0。**printf** 默认输出形式是右对齐、正数前补一个空格、宽度空余部分填充空格。标志位置的附加格式符可以修改默认的格式，其具体含义如表 3-1 所示。

表 3-1 printf 函数常用附加格式符

字符形式	字符含义	
+	正数前的空格改输出为+号	10 输出为+10
-	左对齐，右边空余部分填充空格	ABCD ABC ABCDEF AB
0	宽度空余部分填充 0	10 用 5 个字符宽度输出为：00010

[最小宽度]：十进制整数，表示输出的最少位数。

[.精度]：“.”加上十进制整数 n，其含义是：如果输出的是数值，则该数表示小数位数，若实际小数位数大于该值，则超出部分四舍五入；如果输出的是字符，则表示输出字符的个数。

[长度]：可以是 h、l。h 表示按短整型量输出，l 表示按长整型量或双精度量输出。

类型：是格式说明符中必须要有的，它表示输出列表里要输出的数据类型。表 3-2 给出了常用的类型格式符及含义。

表 3-2　printf 函数常用类型格式符表

格式字符形式	格式字符含义
d	表示以十进制形式输出一个带符号的整数（正数不输出符号）
o	表示以八进制形式输出一个无符号的整数（不输出前导符 0）
x	表示以十六进制形式输出一个无符号的整数
u	表示以十进制形式输出一个无符号的整数
f	表示以小数形式输出带符号的实数（包括单、双精度）
e	表示以指数形式输出带符号的实数
g	表示选择%f 或%e 格式输出实数（选择占宽度较小的一种格式输出）
c	表示输出一个单字符
s	表示输出一个字符串

注意：以上格式说明中精度选项优先于宽度选项，宽度选项是非强制执行的，当遇到实际数据长度超过宽度设定时，宽度选项无效。

下面的例子将逐步演示以上格式说明。

【例 3-4】分析下面程序运行结果。

```
#include <stdio.h>
void main()
{
  char c='A';
  int a = 65 , b = -100;
  float   x = 3.141592631415,y = -3141592631.415;
  double dx = 3.141592631415;
  printf("c=%d, c=%c, c=%x\n",c,c,c);
  printf("a=%d, a=%x, a=%o,a=%c\n",a,a,a,a);
  printf("a=%d, a=%10d,a=%-10d, a=%+d\n",a,a,a,a);
  printf("b=%d, b=%10d,b=%-10d, b=%+d\n",b,b,b,b);

  printf("x=%f,x=%6.f,x=%.3f,x=%6.3f,x=%10.3f\n",x,x,x,x,x);
  printf("y=%f,y=%6.f,y=%10.f\n",y,y,y);

  printf("dx=%f,dx=%6.f,dx=%.3f,dx=%6.3f,dx=%10.3f\n",dx,dx,dx,dx,dx);
```

```
    printf("x=%.8f,dx=%.8f\n",x,dx);

}
```

程序在 TC 和 VC 下运行结果分别如图 3-4 和图 3-5 所示。

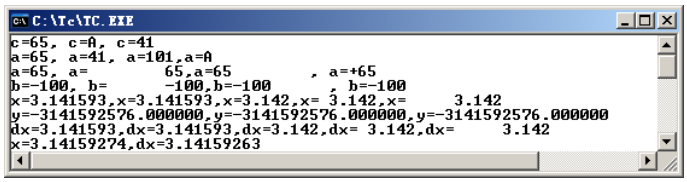

图 3-4　例 3-4 在 TC 的运行结果

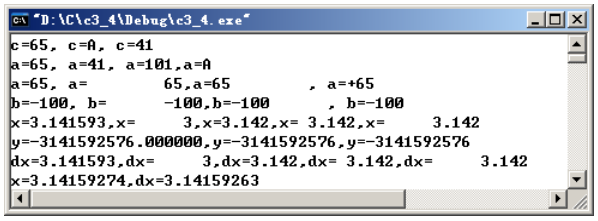

图 3-5　例 3-4 在 VC 下的运行结果

【分析】

（1）char 型变量 c。分别用%d、%c、%x 输出，结果分别为 65、A、41。

（2）int 型变量 a。当输出宽度大于其自身宽度 2 时，空余部分填充空格，附加字符“-”可以将默认的右对齐格式改成左对齐格式，附加字符“+”在正数 65 前加上符号“+”。

（3）负数 b。负数的符号位必须存在，默认比正数多出一个字符位置。

（4）float 型变量 x。TC 下“%f”和“%6.f”输出 x 时意义相同，但在 VC 下，“%6.f”相当于“%6.0f”。x 的精度可以从输出结果中看出，用“%.8f”输出 x 时，其精度只能达到 3.141592，后面的数字是不可知的。

注意：

在使用 **printf** 函数时，要注意以下几个问题：

① 可以在格式控制字符串中包含前面所讲的“转义字符”，如'\n'、'\t'、'\r'、'\b'、'\377'等。

② 跟在%后面的格式符除 X（表示输出的十六进制数用大写字母输出）、E（表示输出的指数 e 用大写字母 E 输出）、G（表示若选用指数形式输出，则用大写字母 E 输出）外，其余必须是小写字母。如%d 不能写成%D。

③ 若想输出字符“%”，则在格式字符串中用连续两个%表示。如：

printf("%f%%",1.0/4);

则输出：0.250000%。

3.4.2　格式化输入函数 scanf

scanf 函数的功能是从键盘上将数据按用户指定的格式输入并赋给指定的变量。

1．scanf 函数调用形式

scanf 函数是一个标准库函数，其调用的一般形式为：

scanf(格式控制字符串,地址列表);

其中格式控制字符串的定义与使用方法和 **printf** 函数相同，但不能显示非格式字符。地址列表是要赋值的各变量地址。地址是由地址运算符“&”后跟变量名组成，如&x 表示变量 x 的地址。“&”是取地址运算符，其作用是求变量的地址。

【例 3-5】**scanf** 函数的使用。

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

运行时按以下方式输入 a、b 的值：

```
100 -200
```

程序将输出：

```
a=100,b=-200
```

输入的 100 和-200 之间有空格。**scanf** 函数的作用是：按照 a、b 在内存中的地址将 100、-200 的值分别存入。数据输入需要分隔，否则无法分辨，默认的分隔符有空格、回车符、Tab（跳格）键。下面的输入方法也是正确的：

- 100□□-200↙　　（用空格“□”作为分隔符）
- 100↙　　（用回车键作为分隔符）
  -200↙
- 100（按 Tab 键）-200↙　（用 Tab 键作为分隔符）

也可以自定义分隔符，例如：

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

输入数据的时候只能按下面的方式：

```
100,-200
```

自定义分隔符“,”也需要输入。

2．格式说明

与 **printf** 函数中的格式说明符相似，以%开始，后面跟一个格式符，中间可以有若干个附加字符，格式字符串的一般形式为：

```
%[*][宽度][长度] 类型
```

【说明】

[ ]：表示可选项。

[*]：表示输入的数值不赋给相应的变量，即跳过该数据不读。

[宽度]：十进制正整数，表示输入数据的最大宽度。

[长度]：长度格式符为 l 和 h，l 表示输入长整型数据或双精度实型数据；h 表示输入短整型数据。

类型：是格式说明符中必须要有的，其格式符的意义与 **printf** 函数基本相同，具体如表 3-3 所示。

表 3-3　scanf 函数常用类型格式符

格式字符形式	格式字符含义
d	表示以十进制形式输入一个整数
o	表示以八进制形式输入一个整数
x	表示以十六进制形式输入一个整数
u	表示以十进制形式输入一个无符号的整数
f 或 e	表示输入一个实数，可以是小数形式或指数形式
g	与 f 或 e 的作用相同
c	表示输入一个字符
s	表示输入一个字符串

【例 3-6】分析下面程序。

```
#include <stdio.h>
void main()
{
  char c;
  int a,b;
  float x,y;
  double dx,dy;
  printf("1.Input a,b(100 -200):");
  scanf("%d%d",&a,&b);                    /*默认空格作为分隔符*/
  printf("a=%d,b=%d\n",a,b);

  printf("2.Input a,b(100,-200):");
  scanf("%d,%d",&a,&b);                   /*自定义逗号“,”作为分隔符*/
  printf("a=%d,b=%d\n",a,b);

  printf("3.Input a,b,c(100 -200A):");
  scanf("%d%d%c",&a,&b,&c);               /*输入字符给变量 c 时，前面不能使用分隔符，因为分隔
符也是字符，所以直接在-200 后面输入字符 A */
  printf("a=%d,b=%d,c=%c\n",a,b,c);

  printf("4.Input a,b,c(100,-200,9:");
  scanf("%d,%d,%c",&a,&b,&c);             /*为了避免默认分隔符可以被字符型变量（%c 可以接受所有
字符，包括转义字符）接收，采用自定义分隔符*/
  printf("a=%d,b=%d,c=%c\n",a,b,c);

  printf("5.Input a,c,b(100A-200):");
  scanf("%d%c%d",&a,&c,&b);               /*不用分隔符的情况。由于字符 A 区别于数字字符，所以系
统可以识别并分隔数据*/
  printf("a=%d,b=%d,c=%c\n",a,b,c);

  printf("6.Input a,b(1112222):");
  scanf("%3d%4d",&a,&b);                  /*采用长度限制，3 个数字字符 111 和 4 个数字字符 2222 分
```

```
别输入给变量 a 和 b*/
  printf("a=%d,b=%d\n",a,b);

  printf("7.Input a,b(1112223333):");
  scanf("%3d%*3d%4d",&a,&b);          /*%*3d 是一种虚读格式，111 后面 3 个字符 222 被读入但没有
赋值给任何变量*/
  printf("a=%d,b=%d\n",a,b);

  printf("8.Input x,y(3.1415926 31415926):");
  scanf("%f%f",&x,&y);                /* float 类型的数据输入，显然所能接收的数据精度只有 7 位，
后面的数字四舍五入*/
  printf("x=%f,y=%f\n",x,y);

  printf("9.Input dx,dy(3.1415926 31415926):");
  scanf("%lf%lf",&dx,&dy);            /* double 类型的数据输入，但用%f 形式输出，只有 6 位小数
精度，读者可以修改为%.8f 试试，观察其小数位数的情况*/
  printf("dx=%lf,dy=%lf\n",dx,dy);

  printf("10.Input x,c,y(3.1415926A31415926):");
  scanf("%f,%c,%f",&x,&c,&y);         /*在输入两个实型数据中间插入一个字符输入，系统可以识别
并分隔数据*/
  printf("x=%f,y=%f,c=%c\n",x,y,c);
}
```

则程序的运行结果如图 3-6 所示。

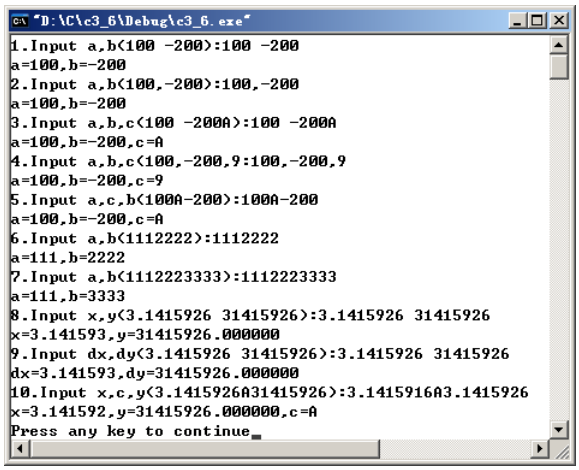

图 3-6　例 3-6 的运行结果

为了调试程序的方便，程序中用 **printf** 语句输出每一步输入操作的提示，括号中是要输入的数据及其格式。例如：

1.Input a,b(100 –200):

请按照提示在冒号后面输入 100-200 即可。

在实际调试的时候，除了变量声明部分外，其他 10 个输出部分可选择性调试。

【分析】在使用 **scanf** 函数时，要注意以下几个问题：

（1）**scanf** 函数中的“格式控制字符串”后面的输入项应该是地址，而不应是变量名。这是 C 语言与其他高级语言不同的地方。例如：

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

不能将语句写成：

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

（2）**scanf** 不支持输入精度控制。例如：

scanf("%8.3f ",&x);

是不合法的。

（3）在“格式控制字符串”中除了格式说明符外，也允许出现其他字符，但在输入数据时在对应位置上应输入与这些字符相同的字符。例如：

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

则输入时应输入：

a=12,b=-2

（4）输入数据时，遇到以下情况认为该数据输入结束：

- 按指定的宽度结束。
- 遇空格，或回车键，或 Tab 键。
- 遇非法输入。如例 3-6 第 5 部分：

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

之所以输入：

100A-200

可以分别使得 a、b、c 为 100、-200、'A'，其主要原因是读入 100 后遇到字符 A，不是数字字符，从而变量 a 的输入结束。

（5）当输入的数据与输出的类型不一样时，虽然编译没有提示出错，但结果将不正确。

3.4.3 字符数据的输入与输出

字符数据也可以通过字符输入函数 **getchar** 和字符输出函数 **putchar** 实现输入和输出。在使用这两个函数时，程序的头部要加上文件包含命令：

#include <stdio.h>

1．字符输入函数 getchar

字符输入函数 **getchar**()的功能是从标准设备（键盘）上读入一个字符。其调用形式为：

getchar();

该函数没有参数，但一对圆括号不能省略。**getchar**()只能从键盘上接收一个字符。

【例 3-7】字符输入函数的使用。

```
#include <stdio.h>
void main()
{
    char c1,c2;
    c1=getchar();
```

```
    c2=getchar();
    printf("%c,%c\n",c1,c2);
}
```

程序运行时，若输入 ab↙，则程序的运行结果如图 3-7 所示；程序运行时，若输入 a□b<回车>，则程序的运行结果如图 3-8 所示。

```
ab
a,b
```

图 3-7　例 3-7 的运行结果 1

```
a b
a,
```

图 3-8　例 3-7 的运行结果 2

字符'b'没有赋给 c2，实际赋给 c2 的是空格。由上可见，两次输入必须连续，不需要分隔符。

2．字符输出函数 putchar

字符输出函数 **putchar**()的功能是向标准输出设备（显示器）输出一个字符。其一般调用形式为：

```
putchar(c);
```

其中 c 是参数，它可以是整型或字符型变量，也可以是整型或字符型常量。当是整型量时，输出以该数值作为 ASCII 码所对应的字符；当是字符型量时，直接输出字符常量。例如：

```
putchar('A');     /*输出字符 A*/
putchar(65);      /*输出 65 所对应的字符 A*/
putchar('\n');    /*输出换行符*/
```

在例 3-7 最后的花括号“}”前添加语句：

```
putchar(c1);putchar(c2); putchar('\n');
```

则程序的运行结果将变成如图 3-9 所示。

```
ab
a,b
ab
```

图 3-9　例 3-7 的运行结果 3

3.5　案例：简单的数据交换算法

【例 3-8】从键盘上输入两个整数放入变量 a 和 b 中，编程将这两个变量中的数据交换。

【分析】实现两个变量的数据交换有很多办法，最常用的是中间变量法。为了交换 a 和 b，需要一个中间变量，例如 t，算法如下：

```
t=a; a=b; b=t;
```

就像两个人（设 a 和 b）交换座位一样，其中 a 先站起来到一个临时位置（设为 t），另一个人 b 坐到 a 座位上，a 再从 t 位置坐到 b 位置上。

下面的算法是错误的：

```
a=b;b=a;
```

当执行 a=b;后，变量 a 原来的值将被“冲掉”，因为变量在任何时刻只能存储一个值，虽然变量的值可以随时被修改。

如图 3-10 所示是交换算法的示意图。

程序如下：

```
#include <stdio.h>
main()
```

```
{
    int a,b,t;
    a=3;b=5;
    t=a;a=b;b=t;
    printf("a=%d,b=%d\n",a,b);
}
```

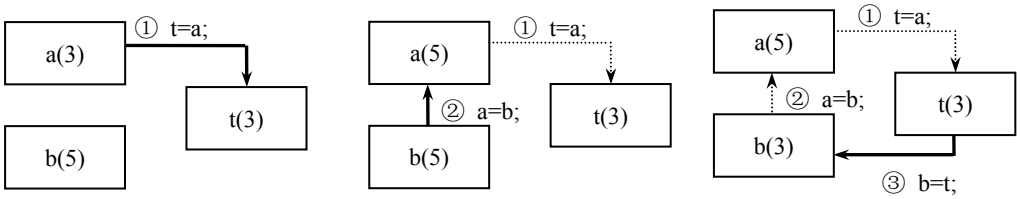

图 3-10　交换算法示意图

程序运行结果如图 3-11 所示。

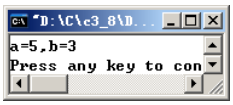

图 3-11　例 3-8 的运行结果

为了方便理解，读者在调试程序的时候也可以输入下面的程序：

```
#include <stdio.h>
void main()
{
  int a,b,t;
  a=3;b=5;
  t=a; printf("a=%d,b=%d,t=%d\n",a,b,t);
  a=b; printf("a=%d,b=%d,t=%d\n",a,b,t);
  b=t; printf("a=%d,b=%d,t=%d\n",a,b,t);
}
```

程序的运行结果如图 3-12 所示。

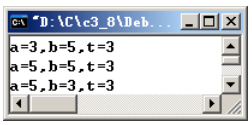

图 3-12　例 3-8 修改后的运行结果

不难发现，t 一直存储着原来 a 的值，所以 b 在 a 的值发生变化后（变成 5）仍然能够获得原先 a 的值 3。

上面的算法也可以写成：

```
t=b;b=a;a=t;
```

道理同上，只不过 b 先发生变化。

3.6　案例：大小写字母的转换

【例 3-9】从键盘上输入一个小写英文字母，编程输出该字母所对应的大写字母。

【分析】大写字母 A～Z 的 ASCII 码值为 65～90，小写字母 a～z 的 ASCII 码值为 97～122。每对字母的 ASCII 码值差都是 32，即'a'-'A'、'b'-'B'、'c'-'C'、…、'z'-'Z'都等于 32。所以将小写字母的 ASCII 码值减去 32，则得到的是所对应的大写字母 ASCII 码值。

程序如下：

```
#include <stdio.h>
void main()
{
  char c1,c2;
  c1=getchar();
  c2=c1 - 32;
  printf("%d,%d,%c,%c\n",c1,c2,c1,c2);
}
```

程序运行时，若输入 a↙，则程序的运行结果如图 3-13 所示。

图 3-13　例 3-9 的运行结果

3.7　案例：计算三角形的面积

【例 3-10】输入三角形的三条边，编程求该三角形的面积。

【分析】三角形面积公式为（设三角形的三条边分别为 a、b、c）：

$$area = \sqrt{s(s-a)(s-b)(s-c)} \qquad 其中\ s = \frac{1}{2}(a+b+c)$$

程序如下：

```
#include <stdio.h>
#include <math.h>
void main()
{
  float a,b,c,s,area;
  scanf("%f%f%f",&a,&b,&c);
  s=(a+b+c)/2;
  area = sqrt(s*(s-a)*(s-b)*(s-c));
  printf("a=%f,b=%f,c=%f\n",a,b,c);
  printf("area=%f\n",area);
}
```

程序运行时，若输入 3.14 4.15 5.16<回车>，则程序的运行结果如图 3-14 所示。

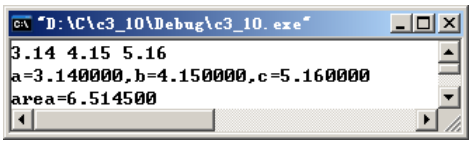

图 3-14　例 3-10 的运行结果

程序中使用了求平方根的函数 sqrt，所以包含了头文件 math.h。

3.8　案例：求一元二次方程的根

【例 3-11】设计程序计算方程的解。其中 a、b、c 用 **scanf** 函数输入（设为 2、3、1）。

【分析】由数学知识可知：求 $ax^2+bx+c=0$ 的根可用求根公式。即当 $b^2-4ac \geq 0$ 时，方程的两个根可用如下公式进行求解：

$$x_{1,2}=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

本题输入的 a、b、c 分别为 2、3、1，则 b^2-4ac 的值为 $3^2-4\times2\times1\geqslant0$，方程的系数满足条件，因此可直接求解。

程序如下：

```
#include <stdio.h>
#include <math.h>
void main()
{
  float a,b,c,d,x1,x2;
  printf("Please input a,b,c:");
  scanf("%f,%f,%f",&a,&b,&c);
  d = sqrt(b*b - 4*a*c);
  x1=(-b+d)/(2*a);
  x2=(-b-d)/(2*a);
  printf("x1=%f, x2=%f\n",x1,x2);
}
```

程序的运行结果如图 3-15 所示。

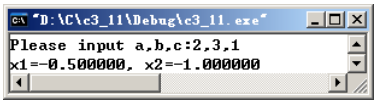

图 3-15　例 3-11 的运行结果

3.9　案例：相同的++运算，不一样的结果

【例 3-12】分析下面程序的运行结果。

```
#include <stdio.h>
void main()
{
  int i,j;
  i=10;
  printf("%d,%d,%d\n",i--,j=i++,j=++i);
  printf("%d,%d\n",i,j);
  printf("%d,%d\n",i+j,j++);
  printf("%d,%d\n",i+j,++j);
  printf("%d\n",(++j+j++));
}
```

本程序在 TC 和 VC 下运行结果不同。左边是 TC 下的运行结果，右边是 VC 下的运行结果，如图 3-16 所示。

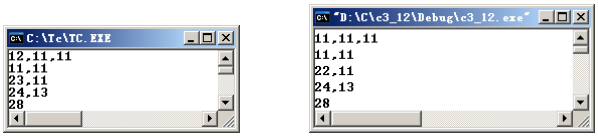

图 3-16　例 3-12 的运行结果

【分析】为了理解程序的执行过程，下面给出一个针对性测试程序及其输出结果（如图 3-17 所示），左边是 TC 下的运行结果，右边是 VC 下的运行结果。

```
#include <stdio.h>
void main()
{
  int i,j,k;
  printf("%d,%d,%d\n",k=j+5,j=i+5,i=10);
  printf("%d,%d,%d\n",k=j++,j=i++,i=10);
  printf("%d,%d,%d\n\n",k=++j,j=++i,i=10);


  printf("%d,%d,%d\n",i+j+5,j=i+5,i=10);
  printf("%d,%d,%d\n",i+j++,j=i++,i=10);
  printf("%d,%d,%d\n",i+++j,j=++i,i=10);
}
```

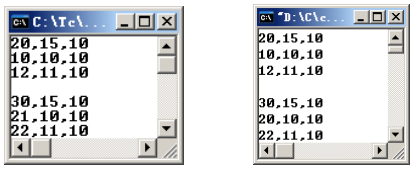

图 3-17　测试程序的运行结果

TC 和 VC 下的输出基本一样，但第五行输出出现了不同。TC 下分别为 21、10、10，而 VC 下分别为 20、10、10。对应的语句是：

```
printf("%d,%d,%d\n",i+j++,j=i++,i=10);
```

注意：

VC 中并没有把 j=i++后缀的影响体现在前面的输出项 i+j++上。这个差别也可以用来解释例 3-12。

读者可以思考以下问题：

如果从第 2 行开始将输出项 i=10 改成 i，结果将会是什么？

本章介绍了顺序程序结构、赋值语句、基本的输入/输出函数。其中重点讲解了以下几方面的内容：

1. 程序结构。C 程序的结构分为顺序结构、选择结构、循环结构。任何 C 程序都由这 3 种结构构成。

2. 赋值语句。由赋值运算表达式构造的赋值语句是最常用的语句，是对变量的最基本操作。

3. 基本的输入/输出函数。

① 格式化输出函数 **printf** 和格式化输入函数 **scanf**。

② 字符输出函数 **putchar** 和字符输入函数 **getchar**。

使用上述函数需要包含头文件 **stdio.h**，TC 中允许在使用 **printf** 和 **scanf** 函数时省略包

含该头文件。

4. 格式字符。

格式字符是以%开头、类型字符结尾的特殊字符串，其中输出格式字符要更为复杂些。

① 输出格式字符。可以简单理解成：% [标志] [宽度] [精度] [长度] 类型。

类型字符是必须有的，主要有 d、o、x、u、f、e、g、c、s，其中 d、o、x、u 用于整型数据，f、e、g 用于实型数据，c、s 用于字符型数据。

- d 表示十进制整数，o 表示八进制整数，x 表示十六进制整数，u 表示无符号整数。
- f 表示十进制小数，e 表示十进制指数，g 表示自动选取 f 或 e 中较短长度的格式。
- c 表示单个字符，s 表示一串字符。
- x、e、g 可以是大写的 X、E、G，相应输出结果中的字母也将是大写。
- e 默认输出 5 位小数，指数部分 TC 下默认 2 位、VC 下默认 4 位。

标志、宽度、精度、长度是可选的，如果没有设置，则按默认的格式输出。

标志有+、-、0 三种字符，“+”用于增加标注正数前面的“+”号，“-”用于更改对齐方式为左对齐，“0”用于修改默认空余填充的空格为“0”。

宽度是十进制整数，不是强制执行的，当宽度小于实际宽度时无效，如果超出，则默认填充空格。当宽度超出实际宽度时，还会带来对齐的问题，默认是右对齐。

精度是十进制整数，主要用于实型和字符型数据的输出控制。实型数默认输出 6 位小数，精度可以修改小数位数，精度不同于有效数字位数。

同时包含宽度和精度的格式串，首先处理精度，然后得到实际宽度，再把设置的宽度和实际宽度比较，超出则填充空格，否则设置宽度无效。

长度有 h、l 两种字符。h 用于标注是短整型，l 用于标注是长整型或 **double** 实型。值得注意的是，32 位机器下，VC 编程环境中 **int** 和 **long** 都是 4 个字节，在输出 **long** 型数据时可以不加 l 修饰了。

② 输入格式字符。可以简单理解成：% [*][宽度] [长度] 类型。

很显然，输入格式字符要简单得多。

宽度、长度和类型的意义基本同输出格式符。“*”表示虚读。

一、选择题

1．若 x、y、z 都定义是 int 类型且初值为 0，则以下不正确的赋值语句是（　　）。

A．x=y=z+10;　　B．x+=y+2;　　C．z++;　　D．x+y+z;

2．下面不是 C 语言语句的是（　　）。

A．int i;　　B．;　　C．a=1,b=5　　D．{ ;}

3．以下合法的 C 语言赋值语句是（　　）。

A．a=b=58　　B．k=a+b　　C．a=58,b=58　　D．- -i;

4．运行下面的程序：

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

```
void main()
{
    int a=5,b=3;
    printf("%d\n",a=a/b);
}
```

则输出结果是（　）。

A．5　　B．1　　C．3　　D．2

5．若变量已正确说明为 int 类型，要给 a、b、c 输入数据，以下正确的输入语句是（　）。

A．scanf("%d%d%d",&a,&b,&c);　　B．scanf("%d%d%d",a,b,c);

C．scanf("%D%D%D",&a,&b,&c);　　D．scanf("%d%d%d",&a;&b;&c);

6．已知 a、b、c 为 float 类型，执行语句：scanf("%f%f%f",&a,&b,&c);使得 a 为 10，b 为 20，c 为 30，则以下不正确的输入形式是（　）。

A．10
20
30

B．10.0,20.0,30.0

C．10.0
30

D．10 20
20.0 30.0

7．若变量已正确定义，现要将 a 和 b 中的数据进行交换，下面语句不正确的是（　）。

A．a=a+b,b=a-b,a=a-b;　　B．t=a,a=b,b=t;

C．a=t; t=b; b=a;　　D．t=b; b=a; a=t;

8．执行下面的程序：

```
#include <stdio.h>
void main()
{
    int a=1,b=2,c=3;
    c=(a+=a+2),(a=b,b+3);
    printf("%d,%d,%d\n",a,b,c);
}
```

则输出结果是（　）。

A．2,2,4　　B．4,2,3　　C．4,2,5　　D．5,5,3

9．执行下面的程序：

```
#include <stdio.h>
void main()
{
    int a;
    float b,c;
    scanf("%2d%3f%4f",&a,&b,&c);
    printf("\na=%d,b=%.1f,c=%.1f\n",a,b,c)
}
```

运行时，从键盘上输入 12345654321↙，则输出结果是（　）。

A．a=12,b=345,c=6543　　B．a=12,b=123,c=1234

C．a=12,b=345.0,c=6543.0　　D．a=12.0,b=345.0,c=6543.0

10．执行下面的程序：

```
#include <stdio.h>
void main()
{
```

```
    int a=3,b=7;
    printf("a=%%d,b=%%d\n",a,b);
}
```

则输出结果是（　　）。

A．a=%3,b=%7　　B．a=%d,b=%d　　C．a=%%d,b=%%d　　D．a=3,b=7

二、阅读程序，写出程序运行结果

1．
```
#include <stdio.h>
void main()
{
    float d, f;
    long k;    int i;
    i=f=k=d=20/3;
    printf("%3d%3ld%5.2f%5.2f \n", i,k,f,d);
}
```

2．
```
#include <stdio.h>
void main()
{
    int x=0177;
    float y=123.4567;
    printf("x=%2d,x=%6d,x=%o,x=%x\n",x,x,x,x);
    printf("y=%8.4f,y=%8.2f,y=%.5f\n",y,y,y);
}
```

3．
```
#include <stdio.h>
void main()
{
    int a=1,b=2;
    a+=b;b=a-b;a-=b;
    printf("%d,%d\n",a,b);
}
```

4．
```
#include <stdio.h>
void main()
{
    int a=1234;
    printf("%2d\n",a);
}
```

5．
```
#include <stdio.h>
void main()
{
    int x=3,y=5;
    printf("%d,%d\n",(x--,--y),x++);
}
```

6．
```
#include <stdio.h>
void main()
```

```
{
    int a=3;
    printf("%d,%d\n",a,(a-=a*a));
}
```

三、程序设计题

1．编程求方程 $2x^2-3x-6=0$ 的根。

2．已知正方体的棱长为 3.2，求正方体的体积和表面积（保留 2 位小数）。

3．输入 3 个整数 a、b、c，编程交换它们的值，即把 a 中的值给 b，把 b 中的值给 c，把 c 中的值给 a。

4．编程将任意输入的小写字母转化成大写字母并输出。