第4章 顺序结构程序设计

为了能编写C语言程序，必须具备以下的知识和能力。

（1）要有正确的解题思路，即学会设计算法，否则无从入手。

（2）掌握C语言的语法，知道怎样使用C语言所提供的功能编写出一个完整的、正确的程序。也就是在设计好算法之后，能用C语言正确表示此算法。

（3）在编写算法和编写程序时，要采用结构化程序设计方法，编写结构化的程序。

本章先从简单的程序开始，介绍简单的算法，同时介绍最基本的语法现象，使读者具有编写简单程序的能力，在此基础上，由浅入深，步步深入。

4.1 顺序程序设计举例

程序流程主要由三种基本结构组成：顺序结构、选择结构和循环结构，顺序结构为程序最基本的结构，其中包含的语句是按书写的顺序执行，且每条语句都执行。

在顺序结构程序中，一般包括以下几个部分。

1．编译预处理命令

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

2．函数

在函数体中，包含顺序执行的各部分语句，主要由以下几部分组成。

（1）变量的说明部分。

（2）提供数据部分。

（3）运算部分。

（4）输出结果部分。

下面介绍几个顺序结构程序设计的例子。

【例4.1】你的一个朋友要去某地旅游，当地天气预报是用华氏温度报告的，请你编写程序，帮助你的朋友进行温度的转换，计算并输出对应的摄氏温度。

1．问题分析

（1）确定预期的输出，问题中有“计算并输出对应的摄氏温度”，明确了输出为摄氏温度celsius。

（2）确定输入项。本需求中，输入项是华氏温度fahrenheit。

（3）列出输入与输出关系的公式。本例中由华氏温度计算摄氏温度的公式为 celsius = 5/9(fahrenheit-32)。

（4）进行手工计算。

2．算法设计

本例中只需要顶层算法：输入一个华氏温度后用公式celsius=5/9(fahrenheit-32)输出摄氏温度值。

程序代码如下：

```
#include<stdio.h>
 int main()
{float Celsius,Fahrenheit
   printf("请输入一个华氏温度");
   scanf("%f",&fahrenheit");
   celsius=5/9(fahrenheit-32);
   printf("华氏温度:%.2");
   return 0;
}
```

【例 4.2】从键盘输入梯形的上底、下底和高，计算梯形的面积。

1．问题分析

根据梯形的面积 s= (a+b)*h/2 设计程序。

（1）定义实型变量 a、b、h、s，分别用于存放上底、下底、高和面积。

（2）调用输入函数，从键盘上输入数据给变量 a、b、c。

（3）运用梯形的面积公式求出面积 s。

（4）输出面积 s。

2．算法设计

程序代码如下：

```
#include <stdio.h>
void main()
{
   float a,b,h,s;
   printf("请输入上底、下底和高：");
   scanf("%f%f%f",&a,&b,&h);          /*输入浮点数赋给变量a,b,h*/
   s=(a+b)*h/2.0;                     /*计算面积，赋给变量s*/
   printf("s=%7.2f \n",s);            /*输出面积，数据共占6个字符宽，2位小数*/
}
```

程序的运行结果如图 4.1 所示。

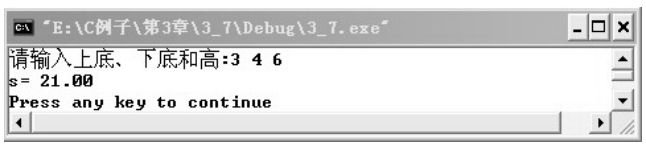

图 4.1　【例 4.2】运行结果

【例 4.3】从键盘输入一个字符，求出与该字符前后相邻的两个字符，按从小到大的顺序输出这三个字符的 ASCII 码。

1．问题分析

ASCII 码的大小关系与字符的大小关系一致，且相邻字符的 ASCII 码编码连续。用顺序结构即可如下设计。

（1）定义字符型变量 ch。

（2）调用字符输入函数，输入一个字符存入变量 ch 中。

（3）分别输入与该字符前后相邻的两个字符及其 ASCII 码。

2．算法设计

程序代码如下：

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

```
    char ch;
    printf("请输入一个字符：");
    ch=getchar();
    printf("%c 的ASCII值为%d\n",ch-1,ch-1);
    printf("%c 的ASCII值为%d\n",ch,ch);
    printf("%c 的ASCII值为%d\n",ch+1,ch+1);
}
```

程序的运行结果如图 4.2 所示。

图 4.2　【例 4.3】运行结果

【例 4.4】输入任意 3 个整数，求它们的平均值。

1．问题分析

（1）定义整型变量 num1、num2、num3 及实型变量 average，分别存放三个整数及平均值。
（2）从键盘上输入 3 个整数存入变量 num1、num2、num3 中。
（3）求 3 个数的平均值，赋给变量 average。
（4）输出 average。

2．算法设计

程序代码如下：

```
#include<stdio.h>
void main()
{
    int num1,num2,num3;
    float average;
    printf("请输入3个整数：\n");
    scanf("%d,%d,%d",&num1,&num2,&num3);      /*输入3个用逗号分隔的整数*/
    average = (num1+num2+num3)/3.0;           /*求平均值*/
    printf("平均值为%8.2f\n",average);        /*输出平均值*/
}
```

程序的运行结果如图 4.3 所示。

图 4.3　【例 4.4】运行结果

思考：为什么求平均值语句是 average = (num1+num2+num3)/3.0，能不能改为 average = (num1+num2+num3)/3？

【例 4.5】 求方程 $ax^2+bx+c=0$ 的实数根($a\neq 0$ 且 $b^2-4ac>0$)。

1．问题分析

一元二次方程的实数根为 $\dfrac{-b\pm\sqrt{b^2-4ac}}{2a}$。

（1）输入实型数据 a、b、c。
（2）求判别式 b^2-4ac。
（3）调用求平方根函数 sqrt()，求方程的根。
（4）输出实数根。

2．算法设计

程序代码如下：

```
#include<stdio.h>
#include<math.h>
void main()
{
    float a,b,c,disc,x1,x2;
    printf("请输入a,b,c:\n");
    scanf("%f,%f,%f",&a,&b,&c);
    disc=b*b-4*a*c;
    x1=(-b+sqrt(disc))/(2*a);
    x2=(-b-sqrt(disc))/(2*a);
    printf("x1=%7.2f   x2=%7.2f\n",x1,x2);
}
```

程序的运行结果如图 4.4 所示。

图 4.4　【例 4.5】运行结果

注意：因为程序中使用了求平方根库函数 sqrt()，因此在预处理命令中应包含#include<math.h>。

4.2　C 语言基本语句

一个 C 程序由数据定义语句及数据处理语句组成，如图 4.5 所示。数据处理语句又可分为表达式语句、函数调用语句、空语句、复合语句、流程控制语句。数据定义语句在第 2 章已经讲解，本章将详细讲解表达式语句、函数调用语句、空语句、复合语句，流程控制语句将在后续章节中讲述。

顺序结构程序由表达式语句、函数调用语句组成，程序流程按语句书写顺序执行。本节简要介绍 C 语言基本语句的相关内容。

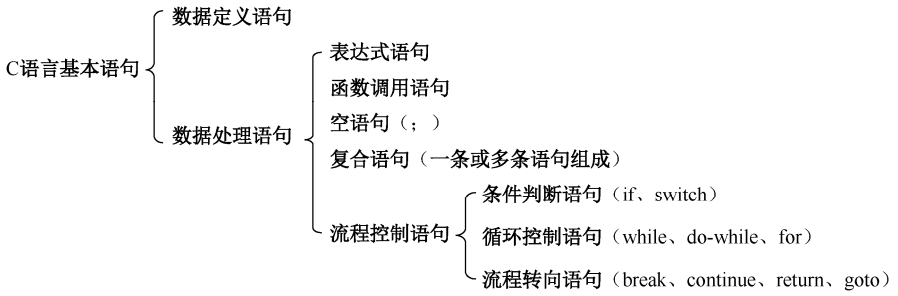

图 4.5　C 语言基本语句

1．表达式语句

由表达式加上“;”组成的语句称为表达式语句。表达式语句的一般形式为：

```
表达式;
```

注意：分号是 C 语句的结束标志。

表达式语句可分为运算符表达式语句和赋值表达式语句，其作用是计算表达式的值或改变变量的值。例如：

```
i++;          /* 自增运算语句，其功能是使变量i的值增1*/
--j;          /* 自减运算语句，其功能是使变量j的值减1*/
y+z;          /* 加法运算语句，但计算结果未保留，无实际意义 */
x=y+z;        /* 赋值语句，先计算y+z的值，然后将此值赋给x */
```

赋值语句是程序中使用最多一种语句，在使用中要注意以下几种情况。

（1）由于在赋值符“=”右边的表达式也可以是一个赋值表达式，因此，下述形式

```
变量=(变量=表达式);
```

是成立的，从而形成嵌套的情形。

其展开之后的一般形式为：

```
变量=变量=…=表达式;
```

例如：

```
a=b=c=d=e=4;
```

按照赋值运算符的右结合性，因此实际上等效于：

```
e=4;
d=e;
c=d;
b=c;
a=b;
```

（2）变量赋初值和赋值语句的区别。

给变量赋初值是属于数据定义语句一部分，赋初值后的变量与其后的其他同类变量之间仍必须用逗号间隔，而赋值语句则必须用分号结尾。例如：

```
int a=4,b,c;
```

在变量说明中，不允许连续给多个变量赋初值。

如下述说明是错误的：

```
int  a=b=c=4;
```

必须写为：

```
int  a=4, b=4, c=4;
```

而赋值语句允许连续赋值。

（3）赋值表达式和赋值语句的区别。

赋值表达式是一种表达式，它可以出现在任何允许表达式出现的地方，而赋值语句则不能。

下述语句是合法的：

```
if  ((x=y+5)>0)  z = x;
```

该语句的功能是，若表达式 x=y+5 大于 0 则 z=x。

下述语句是非法的：

```
if ((x=y+5;)>0)  z=x;
```

因为“x=y+5;”是语句，不能出现在表达式中。

2．函数调用语句

函数调用语句由函数调用表达式后加上“; ”组成，一般形式为：

```
函数名（参数表列）;
```

例如：

```
    pirntf("%d", a);      /* 输出函数调用语句：表示向显示器输出变量a的值 */
    scanf("%d", &a);      /* 输入函数调用语句：表示从键盘输入数据赋给变量a */
```

C 语言有丰富的标准函数库，可提供各类函数供用户调用。标准库函数完成预先设定好的功能，可直接调用。可进行输入输出操作、求数学函数值等。

3．空语句

空语句用一个分号表示，其一般形式为：

```
    ;
```

空语句是什么也不执行的语句。在程序中空语句可用来作为空循环体。

4．复合语句

把多个语句用大括号“{}”括起来组成的一个语句称为复合语句。在程序中应把复合语句看成是单条语句，而不是多条语句。例如：

```
    {
        x=y+z;
        a=b+c;
        printf("%d%d", x, a);
    }
```

是一条复合语句。复合语句内的各条语句都必须以分号“;”结尾，在括号“}”外不能加分号。

5．C 语言数据的输入与输出

输入输出是相对于计算机主机为主体而言的，从计算机向外部输出设备（如显示器、打印机等）输出数据称为“输出”，从外部输入设备（如键盘、鼠标等）向计算机输入数据称为“输入”。

为了让计算机处理各种数据，首先应该把源数据输入到计算机中；计算机处理结束后，再将目标数据信息以人能够识别的方式输出。

C 语言本身不提供输入/输出语句，C 语言的输入/输出操作，是由 C 语言提供的库函数来实现的。如前面所提到的 printf 和 scanf 函数。注意：printf 与 scanf 不是 C 语言的关键字，而只是函数的名字，也可以不用这两个函数名，而重新编写输入/输出函数。

C 语言具有丰富的输入/输出函数，有用于键盘输入和显示器输出的输入/输出函数、磁盘文件读写的输入/输出函数等。本节主要介绍与键盘输入和显示器输出相关的输入/输出函数。

在使用 C 语言库函数时，要用预编译命令“#include”将有关的“头文件”包含在用户源文件中。头文件包含了与用到的函数有关的信息。例如，标准输入/输出函数的相关信息放在“stdio.h”头文件中。因此在调用标准输入/输出函数时要在源文件开头包含以下内容：

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

或

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

stdio 是 standard input and output 的意思。

4.3 字符数据的输入/输出

对于单个字符输入与输出，C 语言标准 I/O 函数库中提供 putchar( )与 getchar( )两个函数。下面简要介绍这两个函数的使用。

4.3.1 字符数据的输出 putchar 函数

putchar 函数是单个字符输出函数，其功能是在标准输出设备上输出单个字符。

其调用的一般格式为：

```
putchar(字符变量);
```

【例 4.6】用 putchar 函数输出数据。

程序代码如下：

```
#include <stdio.h>
void main()
{
    char ch1='B';
    int i=66;
    putchar(ch1);;              /*输出字符变量ch1的值*/
    putchar('\n');              /*换行*/
    putchar(i);                 /*输出字符'B',  字符'B'的ASCII 的值是66*/
    putchar('\n');
    putchar('B');               /*输出字符'B'*/
    putchar('\n');
}
```

程序的运行结果如图 4.6 所示。

使用puthcar函数时要注意以下几点。

（1）putchar 函数可输出一个字符变量、字符常量及整型变量，即将一个整型作为 ASCII 编码，输出相应的字符；也可以是一个转义字符。

（2）putchar 函数只能用于单个字符的输出，且一次只能输出一个字符。

图 4.6　【例 4.6】运行结果

4.3.2　字符数据的输入 getchar 函数

getchar 函数是单个字符输入函数，其功能是从键盘上输入一个字符，并赋值给相应的字符变量或整型变量。

其调用的一般格式为：

```
getchar();
```

【例 4.7】用 getchar()函数输入数据

程序代码如下：

```
#include <stdio.h>
void main()
{
    char  ch;
    printf("请输入两个字符：");
    ch=getchar();                       /*输入一个字符并赋给ch */
    putchar(ch);putchar('\n');
    putchar(getchar());                 /*输入一个字符并输出*/
    putchar('\n');
}
```

程序的运行结果如图 4.7 所示。

（1）getchar()函数只能接收单个字符，输入数字也按字符处理。输入多于一个字符时，只接收第一个字符。

图 4.7 【例 4.7】运行结果

（2）执行 getchar()函数输入字符时，输入字符后需要按 Enter 键后，程序才会响应输入，继续执行后面语句。

（3）getchar()函数也将 Enter 键作为一个回车符读入，因此，在用 getchar()函数连续输入两个字符时要注意回车符。

4.4 格式数据的输入/输出

4.4.1 标准格式输出 printf 函数

printf 函数是格式化输出函数，它的作用是按指定的格式，把指定的数据显示到标准输出设备（显示器）上。

1．printf 函数的一般调用格式

printf 函数的一般调用格式为：

```
printf("格式控制字符串", 输出项表列);
```

例如：

```
printf("x=%d,y=%c\n", x, y);
```

括号内包含以下两部分内容。

（1）格式控制字符串必须用双引号括起来，它的作用是控制输出项的格式和输出一些提示信息，包含以下三种信息。

① 格式控制符，由“%”和格式字符组成。

例如，“%d”表示将输出的数据以整型数据类型输出；“%c”表示将输出的数据以字符类型数据输出。

② 普通字符，这些字符按原样输出。例如，上面例子格式控制字符串中的“x=”、“，”与“y=”都是普通字符。

③ 转义字符，指明特定的操作，如“\n”表示换行，“\t”表示水平制表符。

（2）输出项表列列出要输出的一些数据，如常量、变量、表达式等，它可以是 0 个、1 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。例如：

```
int i = 65;
printf("i=%d, %c\n", i, i);
```

输出结果为：

```
i=65,A
```

语句“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，一个输出 65，另一个输出字符 A，其格式分别由“%d”和“%c”来控制；格式控制字符串中的“i=”是普通字符，按原样输出；“\n”是转义字符，其作用是换行。

2．格式控制符

格式控制符由“%”和格式字符组成，以说明输出数据的类型、形式、长度、小数位数等。

对不同类型的数据用不同的格式控制符。常用的有以下几种格式字符。

（1）d 格式字符。以十进制形式输出整数，主要有以下几种用法：

① %d，按整型数据的实际长度输出。例如：

```
printf ("%d", 1234);
```

输出结果为：

```
1234
```

② %md，m 为指定为输出数据的宽度，即控制输出宽度，为一非负整数。如果数据的位数小于 m，则左端补以空格；若大于 m 位，则按实际位数以十进制数形式输出。例如：

```
int a =123, b = 123456;
printf ("%5d, %5d", a, b);
```

输出结果为：

```
␣␣123,123456              //本教材中用“␣”表示一个空格符
```

%-md，如果数据的位数小于 m，则右端补空格，若大于 m 位，则按实际位数据输出。例如：

```
printf ("-5d, %-5d", a, b);
```

输出结果为：

```
123␣␣,123456
```

③ %ld，输出长整型数据。例如：

```
long a = 123456789;
printf("%ld",a);
```

输出结果为：

```
123456789
```

也可指定数据输出宽度，例如：

```
printf("%10ld",a);
```

输出结果为：

```
␣123456789
```

一个 int 型数据可以用“%d”或“%ld”格式输出。

（2）o 格式字符。以八进制形式输出整数。

输出的数值不带符号，即将符号位也一起作为八进制数的一部分输出。例如：

```
int a = -1;
printf("%d,%o",a,a);
```

在 32 位编译系统中，a 在内存中的存放形式（以补码形式存放）如下：

1	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

因此，其输出结果为：

```
-1,37777777777
```

可以看出，八进制形式输出的整数是不考虑符号的，在使用时要注意。

对长整数（long 型）可以用“%lo”格式输出。同样也可能指定数据的输出宽度。例如：

```
printf("%12o, -12o",a,a);
```

输出结果为：

```
␣ 37777777777,37777777777␣
```

（3）x 或 X 格式字符。以十六进制形式输出整数。

同 o 格式字符一样，十六进制形式输出为整数是不考虑符号的。例如：

```
int a = -1;
printf("%x,%X,%d",a,a,a);
```

输出结果为：

```
ffffffff,FFFFFFFF,-1
```

（4）u 格式字符，以十进制形式输出 unsigned 型数据。例如：

```
int a=-1;
printf("%d,%u",a,a);
```

输出结果为：

```
-1,4294967295
```

（5）c 格式字符。用来输出一个字符。

【例 4.8】字符数据的输出。

程序代码如下：

```
#include <stdio.h>
void main()
{
    char c='b';
    int a=66;
    printf("%c,%d,%3c\n",c,c,c);
    printf("%c,%d,%3c\n",a,a,a);
}
```

图 4.8　【例 4.8】运行结果

程序的运行结果如图 4.8 所示。

一个整数，只要它的值为 0～255，也可以字符形式输出。在输出前，系统会将该整数作为 ASCII 码转换成相应的字符；反之，一个字符也可以作为一个整数输出，输出其所对应 ASCII 码值。

%3c 指定输出的宽度为 3 位，因此左端补 3 个空格。

（6）s 格式字符。用来输出一个字符串，主要有以下几种形式。

① %s，例如：

```
printf("%s","China");
```

输出结果为：

```
China
```

② %ms，表示当字符串长度小于指定的输出宽度 m 时，则在左端补空格；若当长度大于 m 时，则按字符串的实际长度输出。

③ %-ms，表示当字符串长度小于指定的输出宽度 m 时，则在右端补空格；若当长度大于 m 时，则按字符串的实际长度输出。

④ %m.ns，表示输出的字符串占 m 个字符宽度，但只输出字符串中开头的 n 个字符，且字符串右对齐，不足 m 位左边补空格。

【例 4.9】字符串的输出。

程序代码如下：

```
#include <stdio.h>
void main()
{
     printf("%s,%5s,%.4s,%7.2s,%-7.3s\n","Hello", "Hello", "Hello",
"Hello", "Hello");
}
```

程序的运行结果如图 4.9 所示。

其中第 3 个输出项，格式说明为“%.4s”即只指定了 n，没指定 m，自动使 m=n=4 故输出 4 列。第 4 个输出项左边补 3 个空格。第 5 个输出项右边补 2 个空格。

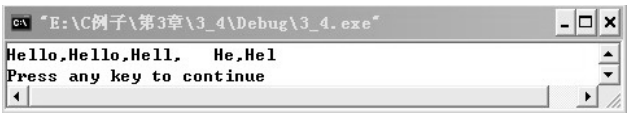

图 4.9　【例 4.9】运行结果

（7）f 格式字符。以小数形式输出十进制实数（包括单、双精度），可以指定格式宽度，也可指定小数位数。主要有以下几种用法。

① %f，不指定宽度，整数部分全部输出，并输出 6 位小数。注意，并非全部数字都是有效数字。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。

② %m.nf，表示输出的实数共占 m 个字符宽度，其中 n 位小数，不足则左端补空格。

③ %-m.nf，与%m.nf 基本相同，只是使输出的数值向右补空格。

【例 4.10】实数的输出。

程序代码如下：

```
#include <stdio.h>
void main()
{
     float x = 123.456;
     double y = 123.456;
     printf("%f,%8f,%8.2f,%.2f,%-8.2f\n",x,x,x,x,x);
     printf("%lf,%8lf,%8.2lf,%.2lf,%-8.2lf\n",y,y,y,y,y);
}
```

程序的运行结果如图 4.10 所示。

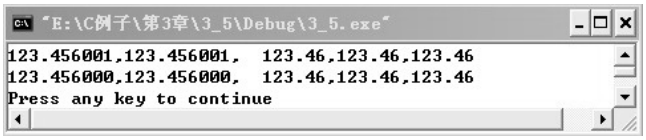

图 4.10　【例 4.10】运行结果

其中第一行的第 3 个输出项左边补 2 个空格，第 4 输出项右边补 2 个空格。

注意：x 的值应为 123.456，而按%f 格式所输出的是 123.456001，这是因为单精度实数的有效位数是 7 位，即只有前面的 7 位是有效数字，数尾的 01 是由于实数在内存中的存储误差引起的。而双精度实数的有效位数是 16 位，所以本例中 y 输出的全是有效数字。

（8）e 格式字符。用来以指数形式输出十进制实数。主要有以下几种用法。

① %e 按标准宽度输出。标准宽度共占 13 位，分别为整数部分占 1 位、小数点占 1 位、小数部分占 6 位、e 占 1 位、指数正（负）号占 1 位、指数占 3 位。

```
printf ("%e",123.4567);
```

输出结果为：

```
1.234567e+002
```

② %m.ne 和%-m.ne。m、n、“-”与前面所讲含义相同。此处 n 表示输出数据的小数部分的小数位数。例如：

```
float f = 321.654;
printf ("%e,%9e,%9.1e,%.1e,%-9.1e",f,f,f,f,f);
```

输出结果为：

```
3.216540e+002,3.216540e+002,␣ 3.2e+002,3.2e+002,3.2e+002␣
```

此外，还有“%g”格式符，用以输出浮点数，它根据数值的大小，自动选取 f 格式或 e 格式。

如果要输出“%”本身，则双写“%”。例如：

```
printf ("%f%%", 1.0/3);
```

输出结果为：

```
0.333333%
```

以上详细介绍了各种格式控制符的使用。下面对格式控制符进行归纳总结。

格式控制符的一般形式为：

```
%[标志][输出最小宽度][ . 精度][长度]类型
```

其中方括号[]中的项为可选项。

各部分说明如下。

（1）标志：标志字符为-、+、#三种，其意义如表 4.1 所示。

表 4.1　标志符及意义

标志	说明
-	结果左对齐，右边填空格
+	输出符号(正号或负号)
#	对 c、s、d、u 类无影响；对 o 类，在输出时加前缀 o；对 x 类，在输出时加前缀 0x；对 e、g、f 类当结果有小数时才给出小数点

（2）输出最小宽度：用十进制整数来表示输出的最少位数。若实际位数多于定义的宽度，则按实际位数输出；若不足则在输出数据的左边或右边补足空格（依据标志符）；若在“输出最小宽度”符前面加前缀“0”，则不足位补 0。

（3）精度：精度格式符以“.”开头，后跟十进制整数。本项的意义是：如果输出整数，则表示至少要输出的数字个数，不足在左边补足数字 0，多则原样输出；如果输出实数，则表示小数的位数；如果输出的是字符，则表示输出字符的个数；若实际位数大于所定义的精度数，则截去超过的部分。

（4）长度：长度格式符为 h 和 l 两种，h 表示按短整型数据输出，l 表示按长整型数据输出。

（5）类型：类型字符用以表示输出数据的类型，其格式字符和意义如表 4.2 所示。

表 4.2　printf 函数格式字符

格式字符	说明
d	以十进制形式输出带符号整数（正数不输出符号）
o	以八进制形式输出无符号整数（不输出前缀 0）
x,X	以十六进制形式输出无符号整数（不输出前缀 0x）
u	以十进制形式输出无符号整数
c	输出单个字符
s	输出字符串
f	以小数形式输出单、双精度实数
e,E	以指数形式输出单、双精度实数
g,G	以%f 或%e 中较短的输出宽度输出单、双精度实数
%	输出百分号（%）

4.4.2　标准格式输入 scanf 函数

scanf 函数是格式化输入函数，它的作用是按指定的格式，从键盘输入数据给指定的变量。

1．scanf 函数的一般调用格式

```
scanf ("格式控制字符串", 地址表列);
```

括号内包括以下两部分内容。

（1）格式控制字符串。与 printf 函数类似，控制字符串可以包含 2 种类型的字符：格式控制符、普通字符。

格式控制符与 printf()函数的相似，普通字符在输入有效数据时，必须原样输入。

（2）地址表列由若干个地址组成的列表，相邻 2 个地址之间，用逗号“,”分开。

地址表列中的地址，可以是变量的首地址，也可以是字符数组名或指针变量。它与格式控制字符串中的格式控制符一一对应。

变量首地址的表示方法：

```
&变量名
```

其中“&”是取地址运算符。

【例 4.11】用 scanf 函数输入数据。

程序代码如下：

```
#include <stdio.h>
void main()
{
    int x, y, z;
    printf("请输入3个数：");
    scanf ("%d%d%d", &x,&y,&z);          /*从键盘上输入x、y、z 的值*/
    printf("%d,%d,%d\n",x,y,z);          /*输出x、y、z 的值*/
}
```

程序的运行结果如图 4.11 所示。

图 4.11　【例 4.11】运行结果

“&x”指变量 x 在内存中的地址。上面 scanf 语句的含义是：从键盘上输入 3 个数分别赋给变量 x、y、z。

“%d%d%d”表示按十进制整数形式输入数据，输入数据时，两个数据之间分隔以一个或多个空格，或回车键 Enter、跳格键 Tab。下面输入均合法：

```
① 2␣␣␣3␣␣␣␣4↙
② 2↙
  3␣␣4↙
③ 2（按Tab键）3↙
  4↙
```

2．格式控制符

与 printf 函数类似，由“%”和格式字符组成。

格式控制符的一般形式为：

```
%[*][输入数据宽度][长度]类型
```

其中有方括号[]的项为可选项。各项的意义如下。

（1）类型：表示输入数据的类型，其格式字符和意义如表 4.3 所示。

表 4.3　scanf 函数格式字符

格式	字符意义
d	输入有符号的十进制整数
o	输入无符号的八进制整数
x	输入无符号的十六进制整数
u	输入无符号的十进制整数
f 或 e	输入实数(可用小数形式或指数形式)
c	输入单个字符
s	输入字符串

（2）“*”符称为输入赋值抑制符：表示该输入项，读入后不赋予相应的变量，即跳过该输入值。例如：

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

当输入为“1 ␣ 2 ␣ 3↙”时，把 1 赋予 a，2 被跳过，3 赋予 b。

（3）输入数据宽度：表示输入项最多可输入的字符个数。如遇空格或不可转换的字符，读入的字符将减少。例如：

```
scanf("%5d",&a);
```

输入“12345678↙”，只把 12345 赋值给变量 a，其余部分被截去。

例如：

```
scanf("%4d%5d%f",&a,&b,&c);
```

输入“200812␣7.1↙”，将把 2008 赋给 a，而把 12 赋给 b，将 7.1 赋给 c。“%4d”控制第一个数据只取 4 个字符，所以将前面 4 个字符转换成整型数 2008；“%5d”控制第二个数据只取后面的 5 个字符，但“12”后是空格，读入的字符将减少，因此只把“12”赋给 b。

（4）长度：长度格式符为 l 和 h，l 表示输入长整型数据(如%ld)和双精度浮点数(如%lf)。h 表示输入短整型数据。

3．使用 scanf 函数的注意事项

（1）scanf 函数中的“格式控制字符串”后面应当是变量地址，而不应是变量名。例如：

```
int a,b;
scanf ("%d,%d", a, b);        /*错误*/
scanf ("%d,%d", &a, &b);      /*正确*/
```

（2）用 scanf 函数输入实数时，对于 float 型变量，格式控制符必须为“%f”；对于 double 型变量，格式控制符必须为“%lf”，否则，会得到不正确的数据。例如：

```
float f;
double e;
scanf ("%f", f);              /*正确 */
scanf ("%lf", f);             /*错误*/
scanf ("%lf", e);             /*正确 */
scanf ("%f",e);               /*错误*/
```

也不允许规定精度。例如：

```
float f;
scanf ("%10.4f", f);  /*错误*/
```

（3）如果输入时类型不匹配，scanf 函数将停止处理。例如：

```
int x, y;
char ch;
scanf ("%d%c%3d", &x, &ch, &y);
```

若输入为“23␣a␣56↙”，则函数将 23 存入赋给 x，空格作为字符赋给 ch，字符'a'作为整型数据读入。

（4）如果在“格式控制字符串”中除了格式字符以外还有其他字符，则在输入数据时应输入与这些字符相同的字符。例如：

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

输入时应用以下形式：

```
a=1,b=2↙
```

习题 4

一、选择题

1．printf 函数中用到格式符“%5s”，其中数字 5 表示输出的字符串占用 5 列，如果字符串长度大于 5，则输出按方式（　　）。

A．从左起输出该字符串，右补空格

B．按原字符长从左向右全部输出

C．右对齐输出该字串，左补空格

D．输出错误信息

2．已有定义“int a= -2;”和输出语句“printf("%8x",a);”，以下正确的叙述是（　　）。

A．整型变量的输出形式只有%d 一种

B．%x 是格式控制符的一种，它可以适用于任何一种类型的数据

C．%x 是格式控制符的一种，其变量的值按十六进制输出，但%8x 是错误的

D．%8x 不是错误的格式控制符，其中数字 8 规定了输出数据的宽度

3．若 x、y 均定义成 int 型，z 定义为 double 型，以下不合法的 scanf 函数调用语句是（　　）。

A．scanf("%d %x %le", &x, &y, &z);

B．scanf("%2d *%d%lf", &x, &y, &z);

C．scanf("%x %*d %o", &x, &y);

D．scanf("%x %o%7.2f", &x, &y, &z);

4．以下说法正确的是（　　）。

A．输入项可以为一个实型常量，如“scanf("%f",3.5)；”

B．只有格式控制字符串，没有输入项，也能进行正确输入，如“scanf("a=%d,b=%d")；”

C．当输入一个实型数据时，格式控制部分应规定小数点后的位数，如“scanf ("%4.2f",&f)；”

D．当输入数据时，必须指明变量的地址，如“scanf("%f",&f)；”

5．以下程序的输出结果是（　　）。

```
#include<stdio.h>
void main( )
{
   int k=17;
   printf("%d,%o,%x\n",k,k,k);
}
```

A．17，021，0x11　　B．17，17，17

B．17，0x11，021　　D．17，21，11

6．下列程序的运行结果是（　　）。

```
#include <stdio.h>
void main()
{
   int a=2,c=5;
   printf("a=%d,b=%d\n",a,c);
}
```

A．a=%2,b=%5　　B．a=2,b=5

C．a=d,b=d　　D．a=2,c=5

7．有如下程序，若要求 a1、a2、c1、c2 的值分别为 10、20、'A'、'B'，正确的数据输入是（　　）。

```
#include<stdio.h>
void main()
{
   int a1,a2;
   char c1,c2;
   scanf("%d%d",&a1,&a2);
   scanf("%c%c",&c1,&c2);
}
```

A．1020AB↙　　B．10 20↙　AB↙

C．10 20　ABC↙　　D．10 20AB↙

8．以下 C 程序正确的运行结果是（　　）。

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

```
void main()
{
    long y=-43456;
    printf("y=%-8ld\n",y);
    printf("y=%-08ld\n",y);
    printf("y=%08ld\n",y);
    printf("y=%+8ld\n",y);
}
```

A．y= -43456
y=- 43456
y=-0043456
y=-43456

B．y=-43456
y=-43456
y=-0043456
y=+ 43456

C．y=-43456
y=-43456
y=-0043456
y= -43456

D．y= -43456
y=-0043456
y=00043456
y=+43456

9．以下 C 程序正确的运行结果是（　　）。

```
#include<stdio.h>
void main()
{
    long y=23456;
    printf("y=%3x\n",y);
    printf("y=%8x\n",y);
    printf("y=%#8x\n",y);
}
```

A．y = 5ba0
y = 5ba0
y = 0x5ba0

B．y = 5ba0
y = 5ba0
y = 0x5ba0

C．y = 5ba0
y = 5ba0
y = 0x5ba0

D．y = 5ba0
y = 5ba0
y = # # # #5ba0

10．阅读以下程序，当输入数据的形式为“25，13，10↙，”正确的输出结果为（　　）。

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

A．x+y+z=48　　B．x+y+z=35
C．x+z=35　　D．不确定值

二、看程序，写运行结果

1．写出下面程序的运行结果_____。

```
#include <stdio.h>
void main()
{
    int x=10;float pi=3.1416;
    printf("%d\n",x);
```

```
    printf("%6d\n",x);
    printf("%f\n",57.1);
    printf("%14f\n",pi);
    printf("%e\n",568.1);
    printf("%14e\n",pi);
    printf("%g\n",pi);
    printf("%12g\n",pi);
}
```

2．写出下面程序的运行结果_____。

```
#include <stdio.h>
void main()
{
    float a=123.456;
    double b=8767.4567;
    printf("%f\n",a);
    printf("%14.3f\n",a);
    printf("%7.4f\n",b);
    printf("%1f\n",b);
    printf("%14.3f\n",b);
    printf("%8.4f\n",b);
    printf("%.4f\n",b);
}
```

3．写出下面程序的运行结果_____。

```
#include <stdio.h>
void main()
{
    int x =7281;
    printf("x=%3d,x=%6d,x=%6o,x=%6x,x=%6u\n",x,x,x,x,x);
    printf("x=%-3d,x=%-6d,x=%$-06d,x=%$06d,x=%%06d\n",x,x,x,x,x);
    printf("x=%+3d,x=%+6d,x=%+08d\n",x,x);
    printf("x=%o,x=%#o\n",x,x);
    printf("x=%x,x=%#x\n",x,x);
}
```

4．写出下面程序的运行结果_____。

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

5．写出下面程序的运行结果_____。

```
#include<stdio.h>
void main()
{
    int i=010,j=10;
    printf("%d,%d\n",++i,j--);
}
```

6．已知字母 A 的 ASCII 码是 65。

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

程序的运行结果_____。

三、填空题

1．在 printf 格式字符中，只能输出一个字符的格式字符是_____；用于输出字符串的格式字符是_____；以小数形式输出实数的格式字符是_____；以标准指数形式输出实数的格式字符是_____。

2．假设变量 a 和 b 为整型，以下语句可以不借助任何变量把 a、b 中的值进行交换。请填空

```
a+=____;b=a-____;a-=____;
```

3．有一输入函数“scanf("%d",k);”则不能使用 float 变量 k 得到正确数值的原因是_____和_____。

scanf 语句的正确形式应该是_____。

4．已有定义“int a;float b,x;char c1,c2;”，为使“a=3,b=7.5,x=12.6,c1= 'a ',c2= 'A '”正确的输入函数调用语句是_____，输入数据的方式是_____。

5．若有以下定义的语句，为使变量 c1 得到字符'A '，变量 c2 得到字符'B '，正确的格式输入形式是_____。

```
char c1,c2;
scanf("%4c%4c",&c1,&c2);
```

6．以下程序的执行结果是_____。

```
#include<stdio.h>
void main()
{
    char c='A'+10;
    printf("c=%c\n",c);
}
```

7．输入任意一个三位数，将其各位数字反序输出（例如输入“123”，输出“321”），请将程序补充完整。

```
#include<stdio.h>
void main()
{
    int x ,a ,b, c ;
    printf("请输入一个三位数：");
    scanf("%d", &x);
    a=x/100;
    _______;
    ___________;
    printf("反序为：%d%d%d",c,b,a);
}
```

8．用 getchar 函数读入两个字符给 c1、c2，然后分别用 putchar 函数和 printf 函数输出这两个字符，请将程序补充完整。

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