第 3 章

程 序 结 构

学习目标

- 熟悉赋值语句的用法。
- 掌握基本输入、输出函数的用法。
- 能够用 C 语言编写基本的顺序结构程序。
- 理解选择结构的概念。
- 掌握用 if 分支选择语句编写程序。
- 掌握 switch 多分支选择语句。
- 掌握使用 while、do-while、for 编写循环语句。
- 掌握循环嵌套的使用。
- 掌握 break 和 continue 语句。

3.1 引言

在第 2 章中对组成 C 语言程序的基本元素：常量、变量、运算符、表达式等因素进行了详细介绍。一个表达式加上一个分号就构成一条 C 语言语句。任何复杂的问题，都可以通过顺序、选择、循环这 3 种基本的程序结构解决。在本章中将详细介绍输入、输出函数的用法，分支选择结构，循环程序结构等。

3.2 C 语句分类

程序对数组的处理是通过执行一系列的语句来实现的，在 C 语言中，语句可以分为以下 5 类。

1. 控制语句

控制语句用于进行程序流程控制，有 9 种，见表 3-1。“()”表示要判断的条件。

2. 函数调用语句

在函数调用表达式后加上一个分号“；”构成函数调用语句。

表 3-1 C 语言中的控制语句

语 句 形 式	功 能
if()…else…	构成选择结构
for()…	构成循环结构
while()…	构成先进行条件判断,再执行循环体的循环结构
do…while()	构成先执行循环体,再进行条件判断的循环结构
continue	结束本次循环
break	终止 switch 或循环语句
switch	进行多路分支选择
goto	无条件跳转程序执行方向
return	从被调用函数返回主调函数

例如:

```
printf("Hello world!\n");
```

3. 表达式语句

在表达式后加上分号“;”构成表达式语句。

例如:

```
sum=1+b;
```

4. 空语句

空语句仅有一个分号,什么也不做。

5. 复合语句

复合语句是用花括号{}括起来的语句序列,例如:

```
int a=2,b=3,t;
if(a>b)
{
    t=a;
    a=b;
    b=t;
}
```

//包含 3 条语句的复合语句
//可以实现 a,b 两个数对调

【学习提示】

(1) C 语言的书写格式比较自由,允许在同一行中写多条语句,也可以将一条语句写在多行上,但是为了清晰的程序结构,尽量不要在同一行中写多条语句。在编写程序时采用缩进格式书写,可以得到清晰的程序结构,但同时也要注意花括号的使用和匹配。

(2) 初学者在编写程序时,容易忽略复合语句的花括号,建议在编写程序时一开始就输入起始花括号和终止花括号,然后再输入其他语句。

3.3　赋值语句

在 C 语言中，=作为赋值运算符，不表示判断。赋值语句是由赋值表达式加上分号构成的表达式语句。一般形式为：

```
变量=表达式;
```

使用赋值语句时，需要注意以下几个方面。

(1) 赋值运算符=右边的表达式也可以是一个赋值表达式，例如：

```
int a,b,c,d;
a=b=c=d=3;
```

按照赋值运算符的右结合规则，该语句等价于：

```
a=3;b=3;c=3;d=3;
```

(2) 赋值语句和赋值表达式是两个完全不同的概念。赋值表达式是一种表达式，可以出现在任何允许表达式出现的地方，而语句则不可以。例如：语句“if(x>y) max=x;”正确，其功能是，如果 x 大于 y，则将 x 的值赋给 max。而语句“if(x>y;) max=x;”就是错误的，因为“x>y;”是语句，不能出现在条件判断表达式中。

【思考题】

若将上述语句改成“if(x>y); max=x;”是否正确？

3.4　基本输入、输出操作的实现

编写程序的目的是要帮助用户处理数据。在程序运行过程中经常需要从输入设备(如键盘)输入数据，再从输出设备(屏幕、打印机等)输出程序的运行结果，从而实现人机交互。

C 语言本身并不提供输入输出语句，其输入输出操作是通过专门的函数实现的。例如，在前面章节案例中用到的 printf()函数和 scanf()函数。在程序中使用这两个函数时，本身就是函数调用，千万不要认为这是 C 语言提供的“输入输出语句”。printf 和 scanf 都不是 C 语言的关键字，它们只是函数名。

由于输入输出函数是系统的库函数，所以在使用时，应在源程序开头添加如下语句：

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

说明：

(1) stdio.h 一般放在程序的开头，因此被称为“头文件”；由于使用了 #include(包含)命令，又被称为“包含文件”。

(2) stdio.h 是 standard input & output 的缩写，在这个头文件中包含了与标准 I/O 库有关的定义。系统提供了很多库函数，它们的定义根据类别不同，放在不同的头文件

中,当需要使用库函数时,需要包含相应的头文件。

(3) C语言的标准库中提供了很多标准输入输出函数,在本节中将重点介绍格式化输入函数scanf(),格式化输出函数printf(),字符输入函数getchar(),字符输出函数putchar()。

3.4.1 格式化输出函数

printf()函数的功能是格式化输出任意数据列表,其一般调用格式为:

printf(格式控制字符串,输出列表);

其中,"格式控制字符串"和"输出列表"是printf()函数的参数。

格式控制的作用是控制输出项的格式,也可以直接输出一些提示信息。"格式控制字符串"必须用双引号引起来,其组成有以下3种形式。

(1) 普通字符:按照原样输出,主要用于输出提示信息。

(2) 转义字符:输出转义字符对应的值,主要用于格式控制。

(3) 格式说明:以%开头,以一个格式符结束,在%和格式控制符之间还可以加入一些附加格式说明符(又被称为修饰符)。格式说明用来对后面的输出列表中的数据进行格式控制,一般形式为:

%[附加格式说明符]格式符

其中,"输出列表"表示要输出的数据,可以是常量、变量、表达式、函数返回值等,每个输出项之间用逗号分隔。该项据实际情况可以省略。

在表3-2和表3-3中列出了printf()函数常用的一些格式控制符和附加格式说明。

表3-2 printf()函数的格式字符

格式符	说 明
d	以带符号的十进制形式输出整数
u	以无符号的十进制形式输出整数
x(或X)	以十六进制无符号形式输出整数(不输出前导字符0x)
o(字母)	以八进制无符号数输出整数(不输出前导字符0)
c	输出一个字符
s	输出字符串
f	以小数形式输出单、双精度数,小数位数6位
e(或E)	以指数形式输出单、双精度数,小数位数6位
g(或G)	自动从%f,%e或%E格式中选择宽度较小的一种使用,不输出无意义的0

表3-3 printf()函数的附加格式说明符

附加格式说明符	说 明
l(字母)	用于长整型,可加在格式符d、o、x、u前面
m(正整数)	数据最小宽度
n(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

【例 3-1】 观察以下程序的运行结果。

```
int i=5;
long j=123;
printf("%d,%3d,%03d,%ld,%-5ld,%05ld\n",i,i,i,j,j,j);
```

程序运行结果：

```
5,  5,005,123,123  ,00123
```

说明：

(1) %3d 表示输出的数字至少占 3 个字符宽。

(2) %03d 表示输出的数字至少占 3 个字符宽，不足 3 位时补 0。

(3) %-5ld 表示输出一个长整型数，至少占 5 个字符宽，并且输出的数字向左靠齐。

【例 3-2】 观察以下语句的运行结果。

```
printf("%o,%x,%X,%u,%d\n",10,10,10,10,10);
```

程序运行结果：

```
12,a,A,10,10
```

说明：

(1) %o 格式控制符，以八进制的形式输出数值，十进制 10 对应的八进制数是 12。

(2) %x 格式控制符，以十六进制的形式输出数值，若是 10 及以上的值，用对应的小写字母进行表示；%X 格式控制符，则以大写字母对大于 9 的数值进行表示。

【例 3-3】 观察以下程序的运行结果。

```
int c=65;
printf("c=%c\n",c);
printf("string:%15s\n","Hello world!");
```

程序运行结果：

```
c=A
string:    Hello world!
```

说明：

(1) %c 格式控制符，以字符的形式输出一个数据。虽然在本例中变量 c 为整数，并赋值为 65，但%c 将该变量的值以字符的形式输出了。值在 0～255 范围之内的整型数据都可以以字符的形式输出，输出结果是对应的 ASCII 字符。

(2) %15s 格式控制符，用于输出字符串。在%与 s 之间加上整数 15，设置输出的字符串至少占 15 个字符宽度。

【例 3-4】 观察以下程序的运行结果。

```
float m;
double n;
m=1111.111111;
```

```
n=2222.222222222;
printf("%f,%f,%e,%g\n",m,n,m,n);
```

程序运行结果：

```
1111.111084,2222.222222,1.111111e+003,2222.22
```

说明：

(1) double类型的精度高于float，float的有效位数是7位，double的有效位数是16位。在float类型的变量中，超出7位的有效数字没有意义。

(2) e格式控制符是以指数形式输出实数。%e输出13位，其中，1位整数、1位小数点、6位小数、5位指数(含字符e和指数的符号)。

(3) g格式控制符根据数值自动从%f或%e中选择合适的精度输出。

3.4.2 格式化输入函数

scanf()函数的作用是格式化输入任意数据列表，其一般调用格式为：

```
scanf(格式控制字符串,地址列表);
```

其中，“格式控制字符串”的使用方法与printf()函数中的类似，“地址列表”是输入信息存放地址的列表，一般是变量地址。表3-4和表3-5分别列出scanf()函数常用的一些格式符和附加的格式说明符。

表3-4 scanf()函数的格式符

格式符	说　明
d,i	用于输入十进制整数
u	以无符号十进制形式输入十进制整数
o(字母)	用于输入八进制整数
x	用于输入十六进制整数
c	用于输入单个字符
s	用于输入字符串(非空格开始，空格结束，字符串变量以'\0'结尾)
f	用于输入实数(小数或指数均可)
Se	与f相同

表3-5 scanf()函数的附加格式说明符

附加格式说明符	说　明
l(字母)	用于长整型数(%ld,%lo,%lx)或double型实数(%lf,%le)
h	用于短整型数(%hd,%ho,%hx)
域宽	指定输入所占列宽
*	表示对应输入量不赋给一个变量

3.4.3 字符输出函数

putchar()函数的功能是向标准输出设备(一般指显示器)输出一个字符，其调用格式为：

```
putchar(ch);
```

其中，ch 为一个字符变量或常量。在一定范围内，整型数据也可以用 putchar()输出。ch 也可以是整型变量或常量。

3.4.4　字符输入函数

getchar()函数的功能是从系统标准输入设备(一般指键盘)输入一个字符，其调用格式为：

```
getchar();
```

该函数的返回值就是从设备得到的字符的 ASCII 值。

【例 3-5】　字符输入输出函数应用举例。

```
#include <stdio.h>
void main()
{
    char ch1,ch2;
    ch1='A';
    ch2=getchar();
    putchar(ch1);
    putchar(ch2);
    putchar(65);
    putchar('\n');
}
```

程序运行结果：

```
M↙
AMA
```

说明：

(1) putchar()函数不仅可以用于输出字符常量和字符变量，也可以将整型变量以字符的形式输出。

(2) putchar('\n')用于输出换行符。

【例 3-6】　已知三角形三边长 a、b、c，计算三角形面积的公式为：

$$S=\frac{1}{2}(a+b+c),\quad \text{area}=\sqrt{S(S-a)(S-b)(S-c)}$$

编写程序，从键盘输入 a、b、c 的值，计算并输出三角形的面积。

```
#include <stdio.h>
#include <math.h>
void main()
{
    float a,b,c;
    float are,s;
    printf("Input a,b,c:");
    scanf("%f,%f,%f",&a,&b,&c);
    s=1.0/2*(a+b+c);
    area=sqrt(s*(s-a)*(s-b)*(s-c));
    printf("area=%.2f\n",area);
}
```

程序运行结果：

```
Input a,b,c:3,4,5↙
area=6.0
```

【常见错误归纳】

(1) 忘记添加头文件。在本例中，除了基本输入、输出函数外，还使用了求平方根的数学函数 sqrt()，该函数的定义在头文件 math.h 中，所以需要添加该头文件。

(2) 编写程序时沿用数学中的书写习惯，把 area=sqrt(s*(s−a)*(s−b)*(s−c))写成 area=sqrt(s(s−a)(s−b)(s−c))，在 C 语言中表示相乘的“*”不能省略。

(3) 忽略了 C 语言中有整除的特性，把数学公式 $S=\frac{1}{2}(a+b+c)$写成如下语句：

```
S=1/2*(a+b+c);
```

3.5 条件语句

3.5.1 if 语句的 3 种形式

if 分支选择语句的语法规则是：先计算所给定的选择条件的值，若其值为真则执行为真的分支，否则执行为假的分支。

if 语句有 3 种形式：单分支选择语句、双分支选择语句、多分支选择语句。

1. 单分支选择语句

单分支选择语句是最基本的条件语句，其一般形式为：

```
if(表达式)
    语句;
```

其流程图如图 3-1 所示，如果表达式的值为真，则执行其后的语句，否则不执行语句。

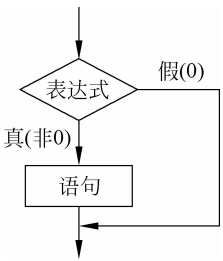

图 3-1 单分支选择语句

【例 3-7】 使用 if 单分支选择语句，输入两个整数，输出其中大的那个数。

```
#include <stdio.h>
void main()
{
    int a,b,max;
    printf("请输入两个数: ");
    scanf("%d%d",&a,&b);
    max=a;
    if(b>max)
        max=b;
    printf("max=%d\n",max);
}
```

程序运行结果：

```
请输入两个数：3 4
max=4
```

2. 双分支选择语句

if-else 构成双分支选择语句，其一般形式为：

```
if(表达式)
    语句 1;
else
    语句 2;
```

该语句的流程图如图 3-2 所示。如果表达式的值为真则执行语句 1，否则执行语句 2。

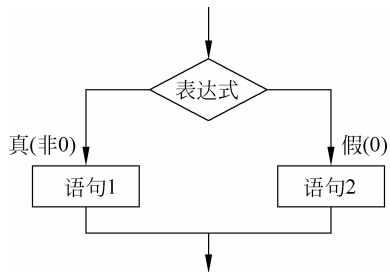

图 3-2　双分支选择语句

【例 3-8】 利用 if 双分支选择语句，输入两个整数，输出其中大的那个数。

```
#include <stdio.h>
void main()
{
    int a,b;
    printf("请输入两个数：");
    scanf("%d%d",&a,&b);
    if(a>b)
        printf("max=%d\n",a);
    else
        printf("max=%d\n",b);
}
```

程序运行结果：

```
请输入两个数：3 4
max=4
```

3. 多分支选择语句

前两种情况的分支选择语句都是用于只有两个分支的情况，if-else if 组成的多分支选择语句可用于多种分支选择的情况，其一般形式为：

```
if(表达式 1)
    语句 1;
else if(表达式 2)
    语句 2;
else if(表达式 3)
    语句 3;
⋮
else if(表达式 n)
    语句 n;
else
    语句 m;
```

其流程图如图 3-3 所示。依次判断每个表达式的值,当某一表达式的值为真时,执行其对应的语句,然后跳到整个 if 之外继续执行程序剩余的部分;若所有的表达式均为假,则执行语句 m,在图 3-3 中,执行语句 4。

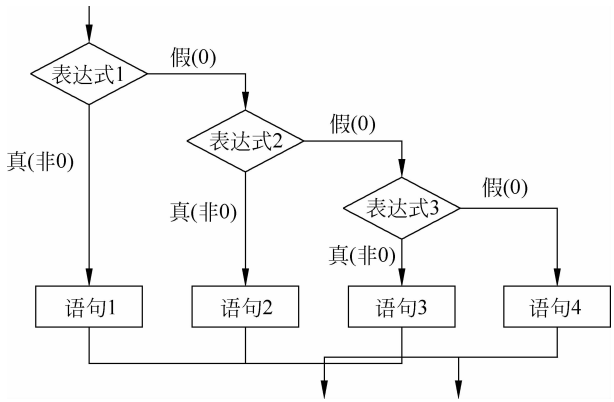

图 3-3 if-else 组成的多分支选择语句

【例 3-9】 利用 if-else 多分支选择语句完成以下题目。学校采用成绩等级和百分制两种体制管理学生成绩,规则如下:成绩在 90～100 之间者为 A,成绩大于 80 而小于 90 的为 B,成绩大于 70 而小于 80 的为 C,成绩大于 60 而小于 70 的为 D,成绩低于 60 的为 F,成绩大于 100 或小于 0 的为无效错误数据。

```
#include <stdio.h>
void main()
{
    int grade;
    printf("请输入一个百分制成绩: ");
    scanf("%d",&grade);
    if(grade>100||grade<0)
        printf("Error\n");
    else if(grade>=90&&grade<=100)
        grade='A';
    else if(grade>=80)
        grade='B';
    else if(grade>=70)
        grade='C';
    else if(grade>=60)
        grade='D';
    else
        grade='F';
    printf("grade=%c\n",grade);
}
```

程序运行结果:

```
请输入一个百分制成绩:
73↙
```