第3章 基本控制结构


视频讲解

3.1 算法与流程基本结构

算法，广义地说，是为解决某一问题而采取的方法和步骤。计算机程序本质上是一个算法，告诉计算机确切的步骤来执行一个指定的任务。因此，算法是指一个被定义好的、计算机可施行其指示的有限步骤或次序，算法包含一系列清晰的指令，并可于有限的时间及空间内清晰地表述出来。

结构化编程方法使用规范的控制流程来组织程序的处理步骤，形成层次清晰、边界分明的结构化构造，每个构造具有单一的入口和出口，从而使程序易于理解、排错、维护和验证正确性。而流程图常用来描述程序的基本操作和控制流程，它是程序分析和过程描述的最基本方式，流程图包含的基本元素如图 3-1 所示。

图 3-1　程序流程图的基本元素

程序流程控制由 3 种基本结构组成：顺序结构、分支结构和循环结构。默认按照程序的书写顺序从上往下顺序执行，即顺序结构，有时也会根据解决问题的需要，采用分支结构和循环结构。分支结构是程序根据条件判断结果选择不同向前执行的一种运行方式，循环结构则是程序根据条件判断结果向后反复执行的一种运行方式。

无论是分支结构还是循环结构，首先都要根据条件判断的结果来选择如何执行，条件判断的结果是个逻辑值，即真或假，通常用关系表达式或逻辑表达式来表示条件判断。

3.2 选择结构

有许多问题有两个以上的可能解，不同的解要通过不同的解题路线得到，由于程序设计时往往难以知道要选哪一条路线，必须同时考虑各种情形，于是就会出现从原问题出发的分支结构，待程序运行时再根据具体情况选择采用哪一条解题路径。选择的依据是某一表达式的值，通常是关系表达式或逻辑表达式的值。

3.2.1 选择语句 if

1. 单分支语句

用 if 语句来表示单分支结构，其形式为：

```
if (表达式)
   语句
```

其中，表达式表示执行条件，如果值为真(非 0)，则执行语句；如果值为假(0)，转向执行后续语句。单分支 if 语句的执行流程如图 3-2 所示。

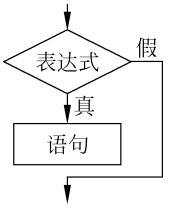

图 3-2 单分支 if 语句的执行流程

例如：

```
if(day == 6||day == 7)
    cout <<"Weekend\n";
```

先计算逻辑表达式 day == 6||day == 7 的值，若为真，则输出 Weekend，否则就跳过该语句。

再如：

```
if ((a + b > c)&&(b + c > a)&&(c + a > b))
{
  s = (a + b + c)/2.0;
  area = sqrt(s * (s - a) * (s - b) * (s - c));
  cout <<"area = "<< area << endl;
}
```

先计算逻辑表达式(a+b>c)&&(b+c>a)&&(c+a>b)的值，若为真，则按顺序执行花括号内的语句块，否则就跳过该语句块。

2. 双分支语句

用 if-else 语句来表示双分支结构，其形式为：

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

其中，表达式表示执行条件，当表达式为真时，执行语句 1；否则，执行语句 2。if-else 语句的执行流程如图 3-3 所示。

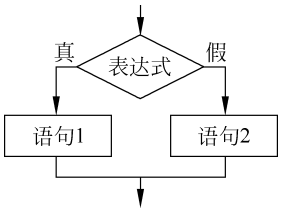

图 3-3 if-else 语句的执行流程

例如：

```
if ((a+b>c)&&(b+c>a)&&(c+a>b))
{
    s = (a+b+c)/2.0;
    area = sqrt(s*(s-a)*(s-b)*(s-c));
    cout<<"area="<<area<<endl;
}
else
    cout<<"It is not a triangle."<<endl;
```

先计算逻辑表达式(a+b>c)&&(b+c>a)&&(c+a>b)的值，若为真，则按顺序执行花括号内的语句块，否则就输出字符串"It is not a triangle."。

【例 3-1】 求两个整数的较大值。

```
#include<iostream>
using namespace std;
int main()
{
    int a,b,max;
    cin>>a>>b;
    if (a>b)
        max=a;
    else
        max=b;
    cout<<"max = "<<max<<endl;
    return 0;
}
```

程序的运行情况及结果如下：

```
3 4↙
max = 4
```

3. 多分支语句

用 if-else 语句来表示多分支结构，其形式为：

```
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 n+1
```

多分支语句的执行流程如图 3-4 所示。执行时先求表达式 1 的值，若表达式 1 的值为真，则执行语句 1；若表达式 1 的值为假，再求表达式 2 的值，若表达式 2 的值为真，则执行语句 2；若表达式 2 的值为假，再求表达式 3 的值，以此类推，若前面 n 个表达式的值均为假，则执行语句 n+1。

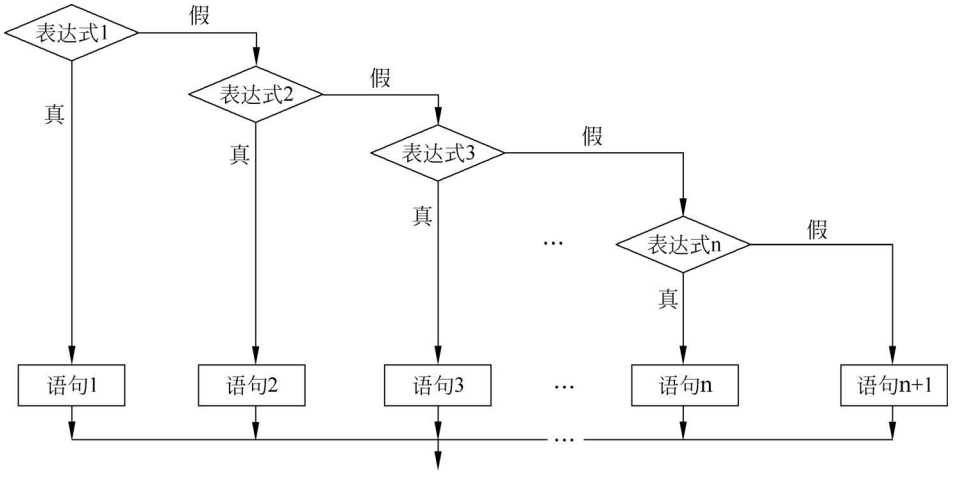

图 3-4　多分支语句的执行流程

【例 3-2】 根据输入的分数情况，输出该分数对应的成绩等级。

```
#include <iostream>
using namespace std;
int main()
{
int score;
cin >> score;
if(score >= 90)
    cout <<"Grade A"<< endl;
else if(score >= 80)                    //分数在 80 与 90 之间
        cout <<"Grade B"<< endl;
else if(score >= 70)                    //分数在 70 与 80 之间
        cout <<"Grade C"<< endl;
else if(score >= 60)                    //分数在 60 与 70 之间
        cout <<"Grade D"<< endl;
else                                    //分数在 60 以下
        cout <<"Grade E"<< endl;
return 0;
}
```

程序的运行情况及结果如下：

```
86↙
Grade B
```

关于 if 语句的 4 点说明如下。

(1) if 后面的表达式可以是符合 C++语法规则的任意表达式，常见的有算术表达式、关系表达式和逻辑表达式。

(2) if 语句可以是单一的语句，也可以是由花括号括起来的复合语句(或称语句块)。

(3) if 和 else 必须配套使用，else 不可单独使用。

(4) if 语句中又可以是 if 语句，这称为 if 语句的嵌套。一般形式如下：

```
if(表达式 1)
    if(表达式 2) 语句 1
    else 语句 2
else
    if(表达式 3) 语句 3
    else 语句 4
```

【例 3-3】　根据 pH 值输出水溶液的酸碱度。

```
#include <iostream>
using namespace std;
int main()
{
double PH;
cin >> PH;
if(PH > 7)
    cout <<"碱性\n";
else
    if(PH < 7)
        cout <<"酸性\n";
    else     //PH 的值为 7
        cout <<"中性\n";
}
```

程序的运行情况及结果如下：

```
7.8↙
碱性
```

注意：if 语句有不同的嵌套形式，但要注意 if 与 else 的配对关系。C++规定，else 总是与它接近的 if 配对。

3.2.2　条件运算符?:

C++还提供了一个三目条件运算符?:用来表示条件表达式，根据逻辑值决定表达式的值。其形式为：

```
操作数 1?操作数 2:操作数 3
```

执行时，先对操作数 1 求值，其值为非 0 时，表达式的值为操作数 2 的值，否则表达式的值为操作数 3 的值。

操作数 1 通常是用于判断的关系表达式或逻辑表达式。例如，表达式 a>b?a:b 的功能是取 a 和 b 中的较大值，因此例 3-1 代码可更改如下：

```
#include <iostream>
using namespace std;
int main()
{
    int x,y,max;
    cin >> x >> y;
    max = x > y?x:y;
    cout <<"max = "<< max << endl;
```

```
    return 0;
}
```

3.2.3 开关语句 switch

switch 语句应用于根据一个整型表达式的不同值决定程序走不同分支的情况，switch 语句的形式为：

```
switch(表达式)
{
    case 常量表达式 1: 语句 1
    case 常量表达式 2: 语句 2
             ⋮
    case 常量表达式 n: 语句 n
    default:           语句 n + 1
}
```

其中，表达式类型为整型、字符型或枚举类型，不能为浮点型。常量表达式具有指定值，与表达式类型相同。

switch 语句的执行流程如图 3-5 所示。先计算表达式的值，然后用这个值依次与 case 后的常量表达式的值进行比较。如果表达式的值等于某个常量表达式 i 的值，则执行语句 i。如果语句 i 之后还有语句，就继续执行语句 i+1 至语句 n+1。如果找不到与表达式的值相等的 case 常量，就执行 default 指示的语句 n+1。

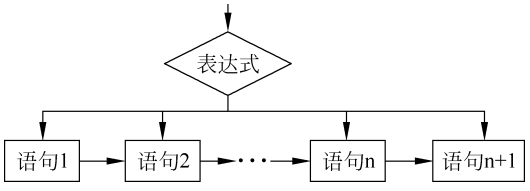

图 3-5　switch 语句的执行流程

break 语句中断一个语句的执行，即能够跳出 switch 语句块，转向执行语句块的后续语句，即 break 可以中止执行 switch 语句中剩下的情况判断和代码执行。break 语句和 default 标号语句均为可选项。

【例 3-4】 测试 switch 语句的执行流程。

```
#include <iostream>
using namespace std;
int main()
{
   int x;
   cout <<"x = ";
   cin >> x;
   switch (x)
   {
   case 1: cout <<"one ";
   case 2: cout <<"two ";
   case 3: cout <<"three ";
```

```
    default: cout <<"other ";
    }
    cout <<"end"<< endl;
    return 0;
}
```

运行程序，输入值为 1，显示结果如下：

```
x = 1↙
one two three other end
```

若重新运行，输入值为 3，显示结果如下：

```
x = 3↙
three other end
```

为了实现真正的选择控制，执行一个 case 标号语句后能跳出 switch 语句块，转向执行后续语句，应该使用 break 语句。

【例 3-5】 测试在 switch 语句中增加 break，中断语句块。

```
#include <iostream>
using namespace std;
int main()
{
    int x;
    cout <<"x = ";
    cin >> x;
    switch (x)
    {
    case 1: cout <<"one ";break;
    case 2: cout <<"two ";break;
    case 3: cout <<"three ";break;
    default: cout <<"other ";
    }
    cout <<"end"<< endl;
    return 0;
}
```

运行程序，输入 x 的值为 1，显示结果如下：

```
x = 1↙
one end
```

选择性地在 case 中使用 break，可以实现多个 case 常量值执行同一个分支语句。

【例 3-6】 两个 case 常量值执行同一个分支语句。

```
#include <iostream>
using namespace std;
int main()
{
```

```
    int x;
    cout <<"x = ";
    cin >> x;
    switch (x)
    {
    case 1:
    case 2: cout <<"one or two ";break;
    case 3: cout <<"three ";break;
    default: cout <<"other ";
    }
    cout <<"end"<< endl;

    return 0;
}
```

程序不管 x 的输入值是 1 还是 2，都执行 case 2 后的语句。输入 1 时，程序执行显示：

```
x = 1↙
one or two end
```

关于 switch 语句需要说明以下几点。

（1）常量表达式必须互不相同，否则就会出现矛盾而引起错误。例如：

```
switch (int(x))
{
    case 1: y = 1; break;
    case 2: y = x; break;
    case 2: y = x * x; break;                    //错误，case 2 已经使用
    case 3: y = x * x * x; break;
}
```

（2）各个 case 和 default 出现的次序可以任意。在每个 case 分支都带有 break 的情况下，case 的顺序不影响执行结果。

（3）switch 语句也可以嵌套。

【例 3-7】 输入年份和月份，输出该月的天数。

```
#include <iostream>
using namespace std;

int main()
{
int year,month,days;
cout <<"year:";cin >> year;
cout <<"month:";cin >> month;
switch(month)
{
case 1:
case 3:
case 5:
case 7:
case 8:
```

```
case 10:
case 12: days = 31; break;
case 4:
case 6:
case 9:
case 11:days = 30; break;
case 2:
    if((year % 4 == 0)&&(year % 100 != 0)||(year % 400 == 0))  //判断当前年是否是闰年
        days = 29;
    else
        days = 28;
}
cout <<"days:"<< days << endl;
return 0;
}
```

程序的运行情况及结果如下：

```
year:2022↙
month:2↙
days:28
```

3.3　循环结构

在一个程序中，常常需要在给定条件成立的情况下，重复地执行某些操作。C++为实现这一目的提供了3种循环语句：while语句、do-while语句和for语句。在循环语句中，重复执行的操作叫作循环体，执行重复操作的条件称为循环条件或循环控制条件。循环体可以是单条语句、多条语句构成的语句块，甚至是空语句。

3.3.1　while语句

while语句的形式为：

```
while（表达式）
    循环体
```

其中，表达式为循环控制条件，循环体用于描述重复执行的一系列操作。

while循环执行流程如图3-6所示。若表达式的值为真，则执行其循环体；否则退出循环，执行while循环后边的语句。while循环又称为当型循环。

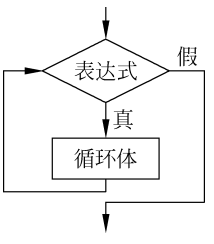

图3-6　while循环的执行流程

从while语句的执行流程可以看到它有以下两个特点。

（1）若条件表达式的值一开始为假，则循环体一次也不执行。

（2）表达式的值为真时，反复执行循环体。为了正常结束循环，循环体内应该包含使得循环条件趋向为假的语句，否则程序将会陷入死循环。

【例 3-8】 用 while 语句求和式 s=1+2+…+100。

```
#include <iostream>
using namespace std;
int main()
{
    int s = 0;
    int i = 1;                          //循环变量的初始化
    while(i <= 100)                     //循环条件
     {                                  //循环体
       s += i;
       i++;                             //改变循环变量
    }
     cout <<"s = 1 + 2 + … + 100 = "<< s << endl;
     return 0;
}
```

该程序的运行情况及结果如下：

```
s = 1 + 2 + … + 100 = 5050
```

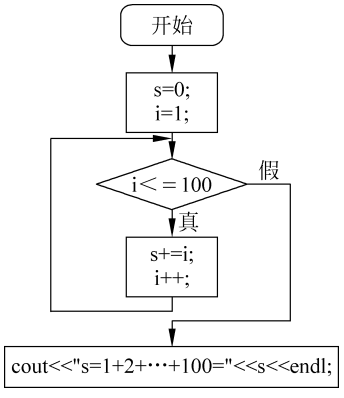

图 3-7 例 3-8 的执行流程

该程序的执行流程如图 3-7 所示。

该例中，变量 s 用来保存多项的和，其被初始化为 0。变量 i 称为循环控制变量，其被初始化为 0。表达式 i<=100 为循环条件表达式，当 i 的值使表达式为真时，不断执行循环体。要想使循环正常结束，循环条件就得为假，那么就需要不断调整 i，调整语句一般应包含在循环体中，如这里的 i++。显然，随着 i 从 0 不断增加到 101 时，循环条件的结果也会从真逐渐变为假，循环也就结束了。

总的来说，一个循环结构包括三部分：循环的初始化、循环条件和循环体。循环开始前要对循环控制变量进行初始化，循环条件通常是包含循环控制变量的一个关系或逻辑表达式，循环体内通常包含对循环控制变量进行调整的语句，使得循环趋向终止。

3.3.2 do-while 语句

do-while 语句的形式为：

```
do
循环体
while (表达式);
```

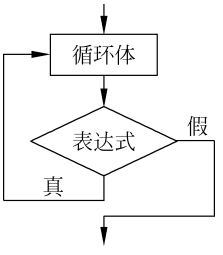

图 3-8 do-while 循环的执行流程

其中，循环体与表达式和 while 语句的含义相同，do-while 语句执行流程如图 3-8 所示。首先执行循环体，然后计算表达式的值，若表达式的值为真，继续执行循环体，否则退出循环，执行 while 循环后边的语句。do-while 循环又称为直到型循环。

与 while 把循环条件放在循环体执行之前不同，do-while 语句把

循环条件判断放在循环体执行之后。从 do-while 语句的执行流程可以看到，不管循环条件是否成立，它都至少执行一次循环体。

【例 3-9】 用 do-while 语句求和式 s＝1＋2＋…＋100。

```
#include <iostream>
using namespace std;
int main()
{
   int s = 0;                          //和初始化为 0
   int i = 1;                          //循环控制变量 i 初始化为 1
   do
   {
     s += i;                           //不断累加
     i++;                              //调整循环控制变量
   }while(i <= 100);                   //分号不能省
   cout <<"s = 1 + 2 + … + 100 = "<< s << endl;
   return 0;
}
```

3.3.3　for 语句

for 语句的一般形式为：

```
for (表达式 1;表达式 2;表达式 3)
    循环体
```

表达式 1 通常用于循环控制变量的初始化；表达式 2 是循环条件表达式即循环条件，其值为真时执行循环，为假时结束循环；表达式 3 通常用于对循环控制变量进行调整，在循环体执行之后执行，也可以在此省略，放在循环体中。其中，表达式 1、表达式 2 和表达式 3 都可以省略。for 语句的执行流程如图 3-9 所示。

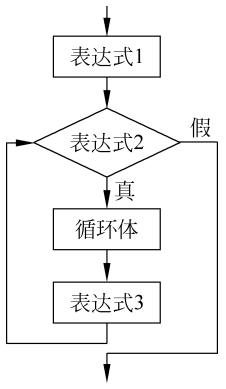

图 3-9　for 语句的执行流程

for 语句的执行过程分析如下：

(1) 求解表达式 1；

(2) 求解表达式 2，若为假，则结束循环，转到(5)；

(3) 若表达式 2 为真，执行循环体，然后求解表达式 3；

(4) 转到(2)；

(5) 执行 for 语句的下一个语句。

从 for 语句的执行过程可以看到，它实际上等效于如下 while 结构：

```
表达式 1;
while(表达式 2)
{
    循环体;
    表达式 3;
}
```

【例 3-10】 用 for 循环求和式 s=1+2+…+100。

```
#include <iostream>
using namespace std;
int main()
{
    int s = 0;
    int i;
    for(i = 1;i <= 100;i++)
        s += i;
    cout <<"s = 1 + 2 + … + 100 = "<< s << endl;
    return 0;
}
```

对于 for 语句的使用需要说明以下几点。

(1) for 语句中省略表达式时，分号不能省略。当省略全部表达式时，for 仅有循环跳转功能。循环变量初始化要在 for 之前设置，而循环条件的判断，循环变量的修改，循环结束控制语句等都要在循环体内实现。

```
for(; ; )
语句块
```

等价于

```
while(1)
语句块
```

例如，求 1+2+…+100 可以写成：

```
int s = 0,i = 1;
for (;;)
{
if (i > 100) break;
s += i;
i++;
}
```

(2) 省略各表达式的 for 语句可以构成不同形式的循环。以下都是求 1+2+…+100 的等价程序。

初始化表达式是逗号表达式，省略第 2 个和第 3 个表达式：

```
for(int s = 0,i = 1;;)
{
if(i > 100)  break;
    s += i;
    i++;
 }
```

省略第 1 个和第 3 个表达式：

```
int s = 0,i = 1;
for(;i <= 100;)
{
```

```
    s += i;
    i++;
}
```

把累加计算表达式放在第 3 个表达式，构成逗号表达式，循环体为空语句。

```
for(int s = 0,i = 1;i <= 100;s += i,i++);
```

读者还可以根据需要和习惯，写出不同形式的 for 循环语句。

【例 3-11】 求斐波那契数列的前 n 项。

算法分析：斐波那契数列形如 0,1,1,2,3,5,8,13,21,…其规律是第 1 项 $a_1=0$，第 2 项 $a_2=1$。从第 3 项开始，每一项都等于前面两项之和。程序中可以使用三个变量 a_1、a_2 和 a_3 进行迭代。

初始时，$a_1=0$，$a_2=1$，根据 a_1 和 a_2 的值计算第 3 项 a_3，即 $a_3=a_1+a_2$；然后用 a_2 的值替换 a_1 的值，用 a_3 的值替换 a_2 的值，用 $a_3=a_1+a_2$ 求得第 4 项的值。如此迭代下去，可以求出斐波那契数列各项的值。

```
#include <iostream>
using namespace std;
int main()
{
    int n,i,a1,a2,a3;
    cout <<"n = ";cin >> n;
    a1 = 0; a2 = 1;                          //对第 1 项和第 2 项赋初始值
    cout << a1 <<'\t'<< a2 <<'\t';
    for(i = 3;i <= n;i++)
    {
        a3 = a1 + a2;                        //求新一项的值
        cout << a3 <<'\t';
        if(i % 8 == 0)                       //每输出 8 项换一行
            cout << endl;
        a1 = a2; a2 = a3;                    //迭代
    }
    return 0;
}
```

程序的运行情况及结果如下：

```
n = 16↙
0       1       1       2       3       5       8       13
21      34      55      89      144     233     377     610
```

3.3.4　循环结构嵌套

循环结构的嵌套，就是在一个循环语句的循环体内又包含循环语句，各种循环语句都可以相互嵌套。一层循环嵌套一层循环称为双层循环，若再嵌套一层循环称为三层循环，根据实际需要可以设计多层循环。

【例 3-12】 双层循环的实现。

```
#include <iostream>
using namespace std;
int main()
{
    cout<<"i\tj\n";
    for(int i=1;i<=3;i++)                    //外循环
    {
        cout<<i;
        for(int j=1;j<=3;j++)                //内循环
            cout<<"\t"<<j<<endl;
    }
    return 0;
}
```

程序的运行情况及结果如下：

```
i       j
1       1
        2
        3
2       1
        2
        3
3       1
        2
        3
```

双层循环的执行过程分析如下。

首先外循环的循环控制变量 i 被初始化为 1，外循环的循环条件 i<=3 的结果为真，转而执行循环体。先输出 1 再执行内循环，内循环的循环控制变量 j 初始化为 1，其循环条件表达式 j<=3 的结果为真，于是执行其循环体，输出 1。然后执行 j++，j 变为 2，循环条件依然为真，接着执行循环体，输出 2。接着执行 j++，j 变为 3，循环条件为真，接着执行循环体，输出 3。再次执行 j++，j 变为 4，循环条件为假，内循环结束。转而计算外循环的第三个表达式 i++，i 变为 2，外循环的循环条件为真，执行其循环体，即先输出 2 再执行内循环。以此类推，直至外循环的循环控制变量变为 4，循环条件为假，此时外循环结束。

【例 3-13】 求 100 以内的所有素数。

算法分析：要判别整数 m 是否为素数，最直接的办法是试除法。即用 2，3，…，m−1 逐个去除 m。若其中没有一个数能整除 m，则 m 为素数；否则，m 不是素数。

数学上可以证明：若所有小于或等于$\sqrt{m}$的数都不能整除 m，则大于$\sqrt{m}$的数也一定不能整除 m。因此，在判别一个数 m 是否为素数时，可以缩小测试范围，只需从 2～$\sqrt{m}$中检查是否存在 m 的约数。只要找到一个约数，就说明不是素数，退出测试。

```
#include <iostream>
#include <cmath>                          //求平方根的函数 sqrt 在头文件 cmath 中定义
```

```
using namespace std;
int main()
{
    int m,i,k,n = 0;
    for(m = 2;m < = 100;m++)
    {
        k = int(sqrt((float)m));            //函数 sqrt 用于返回参数 m 的平方根
        i = 2;
        while(m % i&&i < = k)
            i++;
        if(i > k)
        {
            cout << m <<'\t';
            n += 1;
            if(n % 5 == 0)  cout << endl;    //每输出 5 项换一行
        }
    }
    return 0;
}
```

程序的运行结果如下：

```
2       3       5       7       11
13      17      19      23      29
31      37      41      43      47
53      59      61      67      71
73      79      83      89      97
```

3.4　其他控制语句

当循环语句中的循环条件不满足时，就会结束循环体的执行。而在实际使用中，需要在循环条件仍旧满足的情况下，终止某次循环乃至整个循环。C++为此提供了中断循环的手段。

3.4.1　break 语句

break 语句的形式为：

```
break;
```

前面介绍 switch 语句时，曾使用过 break 语句，它的作用是使某个分支的执行到此结束。除此之外，break 语句还可以用在循环语句中。break 语句的作用是无条件地结束 switch 语句或循环语句，包括 while、do-while 和 for 语句的执行，转向执行语句块的后续语句。

break 语句不能用于 switch 语句和循环语句之外的任何结构语句中。

3.4.2　continue 语句

continue 语句的形式为：

```
continue;
```

continue 语句用于循环体中，终止当前一次循环，不执行 continue 的后续语句，转向循环入口继续执行。

break 语句和 continue 语句的执行流程如图 3-10 所示。

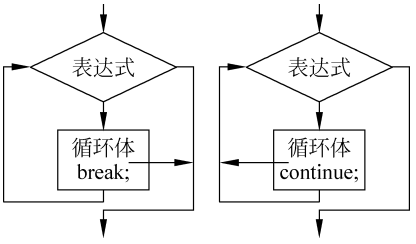

图 3-10　break 语句和 continue 语句的执行流程

【例 3-14】 break 语句与 continue 语句的测试。

```
#include <iostream>
using namespace std;

int main()
{
    int i;
    for(i=10;i<=20;i++)
    {
        if(i%2)
            continue;
        cout<<i<<" ";
    }
    cout<<endl;
    for(i=10;i<=20;i++)
    {
        if(i%2)
            break;
        cout<<i<<" ";
    }
    cout<<endl;
    return 0;
}
```

程序的运行情况及结果如下：

```
10 12 14 16 18 20
10
```

程序中，第 1 个 for 循环输出了 10～20 的所有偶数，第 2 个 for 循环输出了 10～20 的第一个偶数。

3.5　综合举例

【例 3-15】 使用公式 $\frac{\pi}{4}=1-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\cdots$ 求 π 的近似值，直到最后一项的绝对值小于 10^{-6} 为止。

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
    int i = 1,sign = 1;                    //i 表示每一项的分母,sign 表示每一项的符号
    double s = 1,t = 1;                    //s 用来求和,t 表示每一项
    do
    {
        i = i+2;                           //分母每次递增 2
        sign = -sign;                      //符号不断变化
        t = sign * 1.0/i;
        s = s+t;                           //把每一项累加上
    }while(fabs(t)>=1e-6);
    cout<<"PI = "<<4*s<<endl;
        return 0;
}
```

程序的运行情况及结果如下：

```
PI = 3.14159
```

【例 3-16】 求 n!＝1×2×…×n 的值，n 从键盘输入。

```
#include <iostream>
using namespace std;
int main()
{
    int i,n;
    long int t;
    cout<<"n = ";
    cin>>n;
    t = 1;
    for(i=1;i<=n;i++)
        t *= i;
    cout<<n<<"!= "<<t<<endl;
    return 0;
}
```

程序的运行情况及结果如下：

```
n = 4↙
4!= 24
```

【例 3-17】 已知公鸡每只 5 元，母鸡每只 3 元，小鸡每只 1 元。现要用 100 元买 100 只鸡，问公鸡、母鸡、小鸡各为多少？

算法分析：这个问题采用“穷举法”来求解，即是把问题的解的各种可能组合全部罗列出来，并判断每一种可能组合是否满足给定条件，若满足给定条件就是问题的解。

设 x、y 和 z 分别表示公鸡、母鸡和小鸡的数目。由题意可知 x 的取值为 0～19 的整数，y 的取值为 0～33 的整数，所以可以用外层循环控制 x 在 0～19 变化，内层循环控制 y 在

0～33变化，在内层循环中对每一个x和y，求出z，并判断x、y和z是否满足条件：5x＋3y＋z/3＝100，若满足就输出x、y和z。

```
#include <iostream>
using namespace std;
int main()
{
    int x,y,z;
    cout<<"公鸡:\t"<<"母鸡:\t"<<"小鸡:\t"<<endl;
    for(x=0;x<=19;x++)                          //公鸡取值范围
        for(y=0;y<=33;y++)                      //母鸡取值范围
        {
            z = 100-x-y;                        //小鸡数量
            if((5*x+3*y+z/3.0)==100)            //百钱买百鸡
                cout<<x<<'\t'<<y<<'\t'<<z<<endl;
        }
    return 0;
}
```

程序的运行情况及结果如下：

```
公鸡:   母鸡:   小鸡:
0       25      75
4       18      78
8       1       81
12      4       84
```

【例3-18】 输入两个正整数，求出它们的最大公约数。

算法分析：求最大公约数有不同的算法，其中速度较快的是辗转相除法。

该算法描述为：m和n为两个正整数，当m＞n时，m与n的最大公约数等于n与余数r＝m％n的最大公约数；不断更新被除数与除数，那么当余数r＝0时，此时的除数n就是我们要找的最大公约数。

```
#include <iostream>
using namespace std;
int main()
{
    int m,n,a,b,r;
    cout<<"输入两个正整数:"<<endl;
    cout<<"m=";cin>>m;
    cout<<"n=";cin>>n;
    if(m>n)
    {
        a=m;
        b=n;
    }
    else
    {
        a=n;
        b=m;
```

```
    }
    while((r = a % b)!= 0)        //r 表示余数
    {
        a = b;
        b = r;
    }
    cout << m <<"和"<< n <<"的最大公约数为:"<< b << endl;
    return 0;
}
```

运行程序,求 24 和 18 的最大公约数,运行结果如下:

```
输入两个正整数:
m = 24↙
n = 18↙
24 和 18 的最大公约数为:6
```

练习题

一、选择题

1. 已知"int i, x, y;"下列选项中错误的是_____。

A. if(x==y)　i++;　　B. if(x=y)　i--;

C. if(xy)　i--;　　D. if(x+y)　i++;

2. 设有函数关系为 $y=\begin{cases}-1, & x<0\\ 0, & x=0\\ 1, & x>0\end{cases}$,下面选项中能正确表示上述关系的是_____。

A.
```
y = 1;
if (x>=0)
    if (x==0) y = 0;
    else y =-1;
```

B.
```
y = 1;
if (x>=0)
    if (x==0) y = 0;
else y = 1;
```

C.
```
if (x<=0)
    if (x<0)  y = -1;
    else  y = 0;
else  y = 1;
```

D.
```
y = -1;
if (x<=0)
    if (x<0)  y = -1;
    else  y = 1;
```

3. 假设 i = 2,执行下列语句后,i 的值为_____。

```
switch (i)
{
  case 1: i++;
  case 2: i-- ;
  case 3: ++i; break;
  case 4: --i;
  default: i++;
}
```

A. 1　　B. 2　　C. 3　　D. 4

4. 已知“int i = 0, x = 0;”下面 while 语句执行时循环次数为_____。

```
while (!x&&i<3)  {x++; i++;}
```

A. 4　　B. 3　　C. 2　　D. 1

5. 已知“int i = 3;”下面 do-while 语句执行时循环次数为_____。

```
do {i--; cout<<i<<endl;}while(i!=1);
```

A. 1　　B. 2　　C. 3　　D. 无限

6. 下面 for 语句执行时循环次数为_____。

```
for(int i=0,j=5;i=j;)
{
cout<<i<<j<<endl;
i++;j--;
}
```

A. 0　　B. 5　　C. 10　　D. 无限

7. 以下为死循环的程序段是_____。

A. for (int x = 0; x<3;) {x++;};

B. int k = 0;
   do {++k;} while (k>=0);

C. int a = 5; while (a) {a--;};

D. int i = 3; for(; i; i--);

8. 以下程序的输出结果是_____。

```
#include<iostream>
using namespace std;
int main()
{
int i=0,a=0;
while(i<20)
{
    for(;;)
    { if((i%10)==0) break;
        else i--;
    }
    i+=11;
   a+=i;
}
cout<<a<<endl;
return 0;
}
```

A. 11　　B. 32　　C. 21　　D. 33

二、填空题

1. 写出以下程序的输出结果_____。

```
#include <iostream>
using namespace std;
int main()
{
int a,b,c,d,x;
a = c = 0; b = 1; d = 20;
if(a)   d = d - 10;
else if(!b)
         if(!c)   x = 15;
         else   x = 25;
cout << d << endl;
return 0;
}
```

2. 写出以下程序的输出结果_____。

```
#include <iostream>
using namespace std;
int main()
{
int i = 1;
while(i <= 10)
    if(++i % 3!= 1)
        continue;
    else cout << i << endl;
return 0;
}
```

3. 写出以下程序的输出结果_____。

```
#include <iostream>
using namespace std;
int main()
{
int i = 0,j = 5;
do
{
    i++;j -- ;
    if(i > 3)break;
}while(j > 0);
cout <<"i = "<< i <<'\t'<<"j = "<< j << endl;
return 0;
}
```

4. 写出以下程序的输出结果_____。

```
#include <iostream>
using namespace std;
int main()
{
int i,j;
for(i = 1,j = 5;i < j;i++)
{j -- ;}
```

```
cout << i <<'\t'<< j << endl;
return 0;
}
```

5. 写出以下程序的输出结果_____。

```
#include <iostream>
using namespace std;
int main()
{
int i,s = 0;
for(i = 0;i < 5;i++)
switch(i)
{
    case 0:s += i;break;
    case 1:s += i;break;
    case 2:s += i;break;
    default:s += 2;
}
cout <<"s = "<< s << endl;
return 0;
}
```

6. 写出以下程序的输出结果_____。

```
#include <iostream>
using namespace std;
int main()
{
int i;
for(i = 1;i <= 5;i++)
{
    if(i % 2) cout <<"#";
    else continue;
    cout <<"*";
}
cout <<"$\n";
return 0;
}
```

7. 写出以下程序的输出结果_____。

```
#include <iostream>
using namespace std;
int main()
{
int i,j,x = 0;
for(i = 0;i <= 3;i++)
{
    x++;
    for(j = 0;j <= 3;j++)
    {
        if(j % 2)  continue;
```

```
            x++;
        }
        x++;
    }
    cout <<"x = "<< x << endl;
    return 0;
    }
```

三、编程题

1. 输入三角形的三条边长，判断它们能否构成三角形。若能，则判断是等边三角形、等腰三角形还是一般三角形。

2. 我国居民用电实行阶梯电价。阶梯电价是阶梯式递增电价或阶梯式累进电价的简称，是指把户均用电量设置为若干个阶梯分段或分挡定价计算费用。将居民用电量按照满足基本用电需求、正常合理用电需求和较高生活质量用电需求划分为三挡，电价实行分挡递增。总用电量＝第一挡用电量＋第二挡用电量＋第三挡用电量。

试以如下标准根据输入的当年的用电量编程计算当年的电费：

第一挡年用电量 2160 千瓦时：每千瓦时 0.558 元。

第二挡年用电量 2160～4800 千瓦时：每千瓦时 0.608 元。

第三挡年用电量 4800 千瓦时及以上：每千瓦时 0.858 元。

3. 编写程序实现：输入百分制成绩，把它转换成五级分制，转换公式为：

$$\text{grade(级别)}=\begin{cases}\text{excellent(优秀)} & 90\sim100\\ \text{good(良好)} & 80\sim89\\ \text{general(中等)} & 70\sim79\\ \text{pass(合格)} & 60\sim69\\ \text{no pass(不合格)} & 0\sim59\end{cases}$$

4. 已知 XYZ＋YZZ＝532，其中 X、Y 和 Z 为数字，编程求出 X、Y 和 Z。

5. 编程求 1!＋2!＋…＋15!。

6. 编程打印如下图案：

```
            *
          * * *
        * * * * *
      * * * * * * *
    * * * * * * * * *
      * * * * * * *
        * * * * *
          * * *
            *
```

7. 在 100～200 中找出同时满足用 3 除余 2，用 5 除余 3 和用 7 除余 2 的所有整数。

8. 将一个正整数分解质因数，质因数就是一个数的约数，并且是质数。例如：输入 90，输出 90＝2＊3＊3＊5。

9. 输出所有的水仙花数，水仙花数是指一个三位数，其各位数字立方和等于该数本身。

例如，153 是一个水仙花数，因为 $153=1^3+5^3+3^3$。

10. 编程输出 1000 之内的所有完数。所谓完数，是指它的因子之和恰好等于它本身。例如，6 的因子为 1，2，3，而 6＝1＋2＋3，所以 6 是一个完数。

11. 一球从 100 米高处落下，每次落地后反跳回原高度的一半，再落下。编程求它在第 10 次落地时，共经过多少米？第 10 次反弹多高？

12. 用迭代法编程求 $x=\sqrt{a}$。求平方根的迭代公式为：

$$x_{n+1}=\frac{1}{2}\left(x_n+\frac{a}{x_n}\right)$$

要求前后两次求出的 x 的差的绝对值小于 10^{-7}。