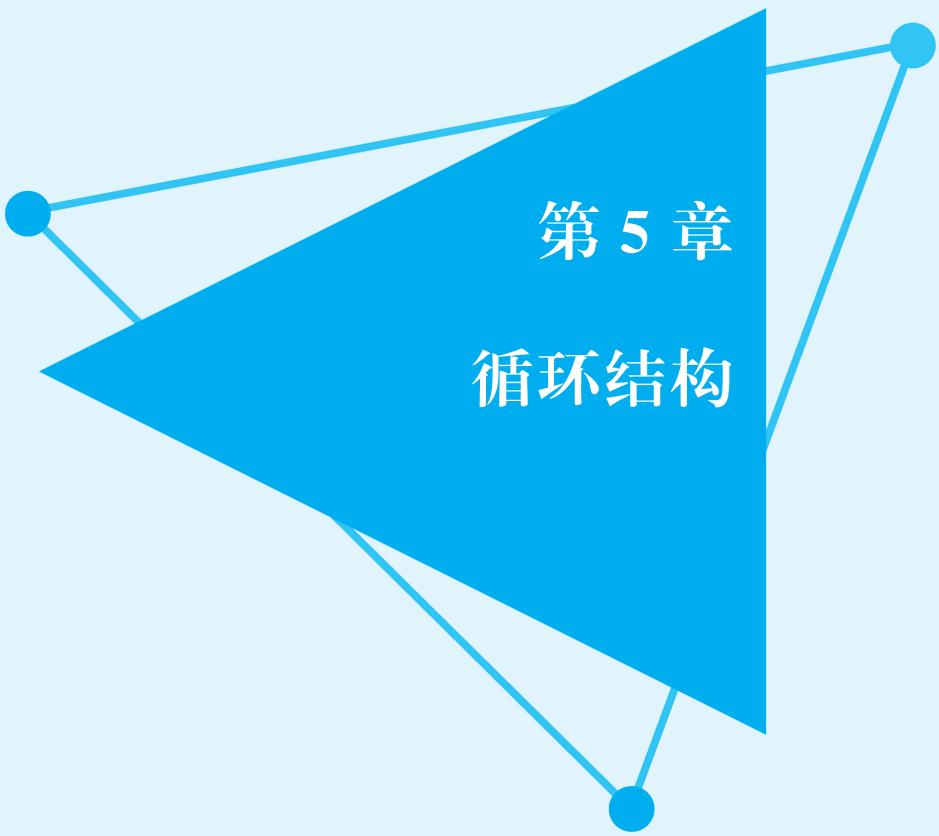

第 5 章

循环结构

在第 1 章中看到算法中有一些步骤是被重复执行的。这种重复执行是通过某一个有条件的跳转指令实现的，即根据某一条件决定某些语句是否被重复执行。这种在程序中不断被重复执行的结构，被称为循环结构。循环结构有时也被称为重复结构。本章主要介绍三种循环结构语句及其应用。

本章学习目标

（1）掌握三种循环语句的基本用法。

（2）能应用循环结构编程解决问题。

（3）能灵活应用穷举法和多重循环。

（4）能灵活使用 break 和 continue 语句。

5.1　循环语句

第 5 章案例代码

1. while 循环（当型循环）

C 语言中，while 语句用来实现“当型”循环结构（图 5-1），它的一般形式如下：

```
while(表达式) 循环体语句;
```

功能说明：

（1）首先求解表达式的值，若表达式的值为真，则执行循环体，否则结束循环。

（2）循环体语句执行完成后，自动转到循环开始处再次求解表达式的值，开始下一次循环。

（3）循环体只能是一个语句。若有多个语句，则应该用大括号将其括起来使之成为一个复合语句。

2. do-while 循环（直到型循环）

在 C 语言中，可以用 do-while 语句实现“直到型”循环结构（图 5-2），它的一般形式如下：

```
do
   循环体语句;
while(表达式);
```

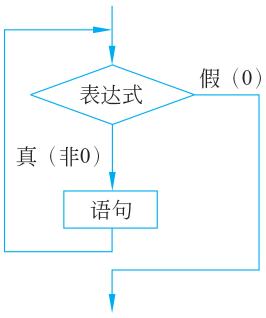

图 5-1　while 循环

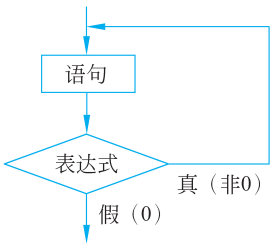

图 5-2　do-while 循环

功能说明：

（1）在此结构中，do 相当于一个标号（其后面不用加：），标志循环结构开始。

（2）首先无条件地执行一次循环体，然后求解表达式的值。若表达式的值为真，则再次执行循环体（转到 do），开始下次循环，否则结束循环。

（3）若循环体多于一个语句，则应使用复合语句。

（4）do-while 结构整体上是一条语句，所以大家不要忘记在 while 的括号后加上语句结束符——分号。

3. for 循环

除了 while 语句和 do-while 语句，C 语言还提供了另外一个使用最为广泛的循环语句——for 语句（图 5-3）。

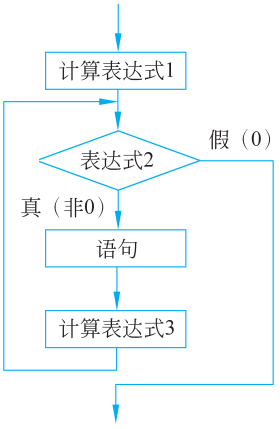

图 5-3 for 语句

for 语句的一般形式为：

```
for(表达式 1;表达式 2;表达式 3) 循环体语句;
```

功能说明：

（1）求解表达式 1（该表达式只在这一步骤被求解一次）。

（2）求解表达式 2，若为真，则执行循环体，否则结束 for 语句。

（3）循环体执行结束后，求解表达式 3，并转向步骤（2）。

（4）循环体语句应该为一条语句，如果有多条语句要执行，则应该使用复合语句。

案例 05-01-01 输出从 1 加到 *n* 的和

案例代码 05-01-01-A.c（while 语句实现）

```
#include<stdio.h>
int main(){
    int i,s,n;
    i=1; s=0;
    scanf("%d",&n);
    while(i<=n){
        s=s+i;
        i=i+1;
    }
    printf("%d",s);
```

```
    return 0;
}
```

执行程序，输入：1000，输出：500500

案例代码 05-01-01-B.c（do-while 语句实现）

```
#include<stdio.h>
int main(){
    int i,s,n;
    scanf("%d",&n);
    i=1; s=0;
    do{
        s=s+i;
        i=i+1;
    }while(i<=n);
    printf("%d",s);
    return 0;
}
```

案例代码 05-01-01-C.c（for 语句实现）

```
#include<stdio.h>
int main(){
    int i,s=0,n;
    scanf("%d",&n);
    for(i=1;i<=n;i++)
        s=s+i;
    printf("%d",s);
    return 0;
}
```

案例拓展 输出约数

编程输入一个正整数，输出它所有的约数。

输入样例：100　　输出样例：1 2 4 5 10 20 25 50 100

案例 05-01-02 输出最大公约数

输入两个数，输出它们的最大公约数。

输入样例：36　24　　输出样例：12

案例代码 05-01-02-A.c （穷举法，用 while 语句实现）

```
#include<stdio.h>
int main(){
    int m,n,i,k;
    scanf("%d%d",&m,&n);
```

```
    i=1;
    while(i<=n&&i<=m){
        if(m%i==0&&n%i==0) k=i;
        i++;
    }
    printf("%d",k);
    return 0;
}
```

执行程序，输入：

```
36 48
```

输出：

```
12
```

案例代码 05-01-02-B.c （穷举法，用 for 语句实现）

```
#include<stdio.h>
int main(){
    int m,n,i,k;
    scanf("%d%d",&m,&n);
    for(i=1;i<=n&&i<=m;i++)
       if(m%i==0&&n%i==0) k=i;
    printf("%d",k);
    return 0;
}
```

案例代码 05-01-02-C.c （辗转相除法，用 while 语句实现）

```
#include<stdio.h>
int main(){
  int m,n,t;
  scanf("%d%d",&m,&n);
  while(m%n!=0){
    t=m%n;
    m=n;
    n=t;
  }
  printf("%d",n);
  return 0;
}
```

案例拓展 输出最小公倍数

输入两个数，输出它们的最小公倍数。

```
输入样例:36  24     输出样例:72
```

案例 05-01-03 素数判断

输入一个正整数，输出其是否为素数。

案例代码 05-01-03-A.c

```
#include<stdio.h>
int main(){
    int n,i,f=1;
    scanf("%d",&n);
    for(i=2;i<n;i++)                                    //判断 n 在区间[2,n-1]是否有约数
      if(n%i==0) f=0;
    if(f==1) printf("YES");
    else     printf("NO");
    return 0;
}
```

执行程序，输入：

```
15
```

输出：

```
NO
```

案例代码 05-01-03-B.c

```
#include<stdio.h>
int main(){
    int n,i,f=1;
    scanf("%d",&n);
    for(i=2;i<=sqrt(n)&&f==1;i++)              //判断 n 在区间[2,sqrt(n)]是否有约数
      if(n%i==0) f=0;
    printf(f==1?"YES":"NO");
    return 0;
}
```

执行程序，输入：

```
17
```

输出：

```
YES
```

案例拓展 $\sqrt{2}$ 的迭代

有一个迭代公式很神奇：$x_n=\sqrt{x_{n-1}+2}$，无论 x 的初值(正数)选得多么大，若干次迭

代之后，都与$\sqrt{2}$无限接近，也就是说，x 序列的极限是$\sqrt{2}$。假设 $x_0=99999999$，编程输入一个正整数 n，输出 x_n 的值(保留 10 位小数)。

```
输入样例 1:16          输出样例 1:x[16]=2.0000000790
输入样例 2:8           输出样例 2:x[8]=2.0051798692
```

案例 05-01-04 用穷举法算阶乘

输入一个正整数 n 的值，编程输出 $n!$（n 的阶乘）。

案例代码 05-01-04-A.c （穷举法，用 for 语句实现）

```
#include<stdio.h>
int main(){
    long long i, n;
    long long fact=1;              /* 将累乘积 fact 初始化为 1 */
    scanf("%lld", &n);
    for(i=1; i<=n; i++)
        fact *= i;                 /* 实现累乘 */
    printf("%lld!=%lld", n, fact);
  return 0;
}
```

执行程序，输入：5

输出：5!=120

案例代码 05-01-04-B.c （穷举法，用 while 语句实现）

```
#include<stdio.h>
int main(){
    long long i, n;
    long long fact=1;              /* 将累乘积 fact 初始化为 1 */
    scanf("%lld", &n);
    i=1;
    while(i<=n){
        fact *= i;                 /* 实现累乘 */
        i++;
    }
    printf("%lld!=%lld", n, fact);
    return 0;
}
```

案例拓展 乘方计算

给出一个整数 a 和一个正整数 n，求乘方 a^n。

输入格式：一行，包含两个整数 a 和 n。$-1000\leqslant a\leqslant 1000$，$1\leqslant n\leqslant 100$。

输出格式：一个整数，即乘方结果。题目保证最终结果的绝对值不超过 1000000。

输入样例:2 3　　输出样例:8

案例 05-01-05 Fibonacci 数列

Fibonacci 数列的通项公式为：$f_n=\begin{cases}1 & (n=1,2)\\ f_{n-1}+f_{n-2} & (n>2)\end{cases}$，编程读入整数 n（$2<n\leqslant 40$），输出 Fibonacci 数列的前 n 项。

案例代码 05-01-05.c

```
#include<stdio.h>
int main(){
    long int f1,f2,f3,n;
    int i;
    scanf("%ld",&n);
    f1=1; f2=1;
    printf("%ld,%ld", f1, f2);
    for(i=3;i<=n;i++ ){
        f3 = f1 + f2;
        f1 = f2;
        f2 = f3;
        printf(",%ld", f3);
    }
    return 0;
}
```

执行程序，输入 20，输出：

```
1,1,2,3,5,8,13,21,34,55,89,144,233,377,610,987,1597,2584,4181,6765
```

案例拓展 计算 e 的近似值

本题要求：编写程序，输入一个较小的实数 deta，利用 $e=1+\frac{1}{1!}+\frac{1}{2!}+\frac{1}{3!}+\cdots+\frac{1}{n!}$ 计算 e 的近似值，直到最后一项的绝对值小于 deta 时为止，输出此时 e 的近似值。

5.2　循环控制语句

1. break 语句

break 语句的一般形式是：

```
break;
```

功能说明：

(1) 在循环体内，当程序执行到 break 语句时，会立即跳出循环结构，即提前结束循

环体。

（2）break 语句通常出现在某个 if 语句的分支中，以实现有条件地结束循环。

（3）break 语句只能用于 switch 结构内部或循环结构内部。

2. continue 语句

continue 语句的一般形式是：

```
continue;
```

功能说明：

（1）在循环体内，当程序执行到 continue 语句时，会立即结束本次循环（跳过循环体，continue 语句后面的部分不执行），接着执行下一次循环。

（2）continue 语句通常出现在某个 if 语句的分支中。

（3）continue 语句只能用于循环结构的内部。

案例 05-02-01 输出从 1 加到 *n* 的和（应用 break 语句）

案例代码 05-02-01-A.c

```
#include<stdio.h>
int main(){
    int i,s,n;
    i=1;
    s=0;
    scanf("%d",&n);
    while(1){
      s=s+i;
      i=i+1;
      if(i>n) break;
    }
    printf("%d",s);
    return 0;
}
```

执行程序，输入 1000，输出：

```
S=500500
```

案例代码 05-02-01-B.c

```
#include<stdio.h>
int main(){
    int i,s,n;
    scanf("%d",&n);
    for(i=1,s=0; ;i++) {
```

```
        if(i>n) break;
        s=s+i;
    }
    printf("%d",s);
    return 0;
}
```

案例拓展　输出最大公约数(应用 break 语句)

编程输入正整数 m 和 n,输出它们的最大公约数,要求在循环中应用 break 语句。

案例 05-02-02　输出 ASCII 码

输入一串字符(以＃字符结束),依次输出每个字符及其 ASCII 码(不包括结束符＃)。

案例代码 05-02-02.c

```
#include<stdio.h>
int main(){
    char c;
    int i=1;
    do{
        scanf("%c",&c);
        if(c=='#') break;
        printf("%c-%d\n",c,c);
    }while(1);
    return 0;
}
```

执行程序,输入:

```
A23#
```

输出:

```
A-65
2-50
3-51
```

案例拓展　输出数字和

输入一串字符(以＃字符结束),输出这串字符中所有数字字符的和。

```
输入样例:ABC123DE4FG#        输出样例:10
```

案例 05-02-03　输出不能被 2、3、5、7 和 13 整除的数

编程输入整数 a 和 b,输出 a、b 之间(包括 a、b 本身)的不能被 2、3、5、7 和 13 整除的数。