> 学习编程语言语法是次要的，思维是主要的。如何把头脑中的想法变成简洁的代码，至关重要。——闫学灿

学习循环语句只需要抓住一点——代码执行顺序！

一、 `while` 循环

可以简单理解为循环版的 `if` 语句。 `if` 语句是判断一次，如果条件成立，则执行后面的语句；`while` 是每次判断，如果条件成立，则执行循环体中的语句，否则停止。

```
#include <iostream>

using namespace std;

int main()
{
    int i = 0;
    while (i < 10)
    {
        cout << i << endl;
        i ++ ;
    }

    return 0;
}
```

练习：求1~100中所有数的立方和。

```
#include <iostream>

using namespace std;

int main()
{
    int i = 1, sum = 0;
    while (i <= 100)
    {
        sum += i * i * i;
        i ++ ;
    }
    cout << sum << endl;
    return 0;
}
```

练习：求斐波那契数列的第 `n` 项。
`f(1) = 1`, `f(2) = 1`, `f(3) = 2`, `f(n) = f(n-1) + f(n-2)`。

```cpp
#include <iostream>

using namespace std;

int main()
{
    int n;
    cin >> n;

    int a = 1, b = 1, i = 1;
    while (i < n)
    {
        int c = a + b;
        a = b;
        b = c;
        i ++ ;
    }

    cout << a << endl;

    return 0;
}
```

死循环：循环永久执行，无法结束。我们要避免写出死循环。

```cpp
#include <iostream>

using namespace std;

int main()
{
    int x = 1;

    while (x == 1) puts("!");

    return 0;
}
```

二、`do while`循环

`do while`循环不常用。
`do while`语句与`while`语句非常相似。唯一的区别是，`do while`语句限制性循环体后检查条件。不管条件的值如何，我们都要至少执行一次循环。

```cpp
#include <iostream>

using namespace std;

int main()
{
    int x = 1;
    while (x < 1)
    {
        cout << "x!" << endl;
        x ++ ;
    }

    int y = 1;
    do
    {
        cout << "y!" << endl;
    } while (y < 1);

    return 0;
}
```

三、 `for` 循环

基本思想：把控制循环次数的变量从循环体中剥离。

```cpp
for (init-statement : condition: expression)
{
    statement
}
```

`init-statement` 可以是声明语句、表达式、空语句，一般用来初始化循环变量；
`condition` 是条件表达式，和while中的条件表达式作用一样；可以为空，空语句表示 `true` ；
`expression` 一般负责修改循环变量，可以为空。

```cpp
#include <iostream>

using namespace std;

int main()
{
    for (int i = 0; i < 10; i ++ )
    {
        cout << i << endl;
    }

    return 0;
}
```

练习：求1~100中所有数的立方和。
练习：求斐波那契数列的第 `n` 项。
`f(1) = 1` , `f(2) = 1` , `f(3) = 2` , `f(n) = f(n-1) + f(n-2)` 。

`init-statement`可以定义多个变量，`expression`也可以修改多个变量。

例如求`1 * 10 + 2 * 9 + 3 * 8 + 4 * 7 + 5 * 6`：

```
#include <iostream>

using namespace std;

int main()
{
    int sum = 0;
    for (int i = 1, j = 10; i < j; i ++, j -- )
    {
        sum += i * j;
    }

    cout << sum << endl;

    return 0;
}
```

四、跳转语句

1. break

可以提前从循环中退出，一般与`if`语句搭配。
例题：判断一个大于1的数是否是质数：

```
#include <iostream>

using namespace std;

int main()
{
    int n;
    cin >> n;

    bool is_prime = true;
    for (int i = 2; i < n; i ++ )
        if (n % i == 0)
        {
            is_prime = false;
            break;
        }

    if (is_prime) cout << "yes" << endl;
    else cout << "no" << endl;

    return 0;
}
```

2. continue

可以直接跳到当前循环体的结尾。作用与 `if` 语句类似。
例题：求1~100中所有偶数的和。

```
#include <iostream>

using namespace std;

int main()
{
    int sum = 0;

    for (int i = 1; i <= 100; i ++ )
    {
        if (i % 2 == 1) continue;
        sum += i;
    }

    cout << sum << endl;

    return 0;
}
```

五、多层循环

```
#include <iostream>

using namespace std;

int main()
{
    for (int i = 0, k = 1; i < 10; i ++ )
    {
        for (int j = 0; j < 10; j ++, k ++ )
        {
            cout << k << ' ';
        }
        cout << endl;
    }

    return 0;
}
```

练习：打印1~100中的所有质数

```cpp
#include <iostream>

using namespace std;

int main()
{
    for (int i = 2; i <= 100; i ++ )
    {
        bool is_prime = true;
        for (int j = 2; j < i; j ++ )
        {
            if (i % j == 0)
            {
                is_prime = false;
                break;
            }
        }
        if (is_prime) cout << i << endl;
    }

    return 0;
}
```

练习：输入一个 `n` ，打印 `n` 阶菱形。 `n` 是奇数。

`n = 9` 时的结果：

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

```cpp
#include <iostream>

using namespace std;

int main()
{
    int n;
    cin >> n;

    int cx = n / 2, cy = n / 2;

    for (int i = 0; i < n; i ++ )
    {
        for (int j = 0; j < n; j ++ )
            if (abs(i - cx) + abs(j - cy) <= n / 2)
                cout << '*';
            else cout << ' ';
        cout << endl;
    }

    return 0;
}
```