# 【济南中心】JAVA 编程阶梯：基础篇之第四章

## 循环结构

循环语句顾名思义就是当满足循环条件的情况下，反复执行某一段代码，这段被重复执行的代码被称为循环体语句，当反复把循环判断执行这个循环体时，需要在合适的时条件修改为 false，从而结束循环，否则循环将一直执行下去，形成死循环。

**循环结构的分类**

一般我们将循环结构分为三种： for 循环，while 循环，以及 do...while()循环

现在我们分别讲解。

**for 循环：**

**格式**

```
for(初始化表达式;条件表达式;循环后的操作表达式) {
        循环体;
    }
```

复制代码

**执行流程**

a:执行初始化语句

b:执行判断条件语句,看其返回值是 true 还是 false

如果是 true，就继续执行

如果是 false，就结束循环

c:执行循环体语句;

d:执行循环后的操作表达式

e:回到 B 继续。

**举个栗子**

```
/**
*打印数字 0-99
**/
class Demo
{
	public static void main(String[] args){
		for(int x = 0; x < 100; x++){
			System.out.println("x="+x );
		}
	}
}
```

复制代码

**注意事项**

a:判断条件语句无论简单还是复杂结果是 boolean 类型。

b:循环体语句如果是一条语句，大括号可以省略；如果是多条语句，大括号不能省略。建议永远不要省略。

c:一般来说：有左大括号就没有分号，有分号就没有左大括号

**while 循环：**

**格式**

```
while 循环的基本格式:
  while(判断条件语句) {
  循环体语句;
  }
完整格式:
```

```
初始化语句;
  while(判断条件语句) {
  循环体语句;
  控制条件语句;
}

```

复制代码

**执行流程**

a:执行初始化语句 z

b:执行判断条件语句,看其返回值是 true 还是 false

如果是 true，就继续执行

如果是 false，就结束循环

c:执行循环体语句;

d:执行控制条件语句

e:回到 B 继续。

**举个栗子**

```
/**
* 求 1-10 的和
*/
public class Demo {
public static void main(String[] args) {
                int x = 0, sum = 0;
                while (x < 11) {
                        sum += x;
                        x++;
                }
                System.out.println(sum);
        }
}
```

复制代码

## do...while 循环格式

```
1.  do {
2.  循环体语句;
3.  }while(判断条件语句);
4.
5.  完整格式;
6.  初始化语句;
7.  do {
8.  循环体语句;
9.  控制条件语句;
10. }while(判断条件语句);
```

复制代码

### 执行流程

a:执行初始化语句

b:执行循环体语句;

c:执行控制条件语句

d:执行判断条件语句,看其返回值是 true 还是 false

如果是 true，就继续执行

如果是 false，就结束循环

e:回到 b 继续。

## 举个栗子

```
1.  /**
2.  * 打印 1-10;
3.  */
```

```
class Demo {
        public static void main(String[] args) {
                int x = 1;
                do {
                        System.out.println("x=" + x);
                        x++;
                } while (x <= 10);
        }
}
```

复制代码

**循环结构三种循环语句的区别**

三种循环语句其实都可以完成一样的功能，也就是说可以等价转换，但还是有小区别的:

do...while 循环至少会执行一次循环体。

for 循环和 while 循环只有在条件成立的时候才会去执行循环体

**注意事项：**

写程序优先考虑 for 循环，再考虑 while 循环，最后考虑 do...while 循环。

如下代码是死循环

```
while(true){}
for(;;){}
```

复制代码

**for 循环和 while 循环的区别**：for 循环语句和 while 循环语句可以等价转换，但还是有些小区别的

**使用区别：**

控制条件语句所控制的那个变量，在 for 循环结束后，就不能再被访问到了，而 while 循环结束还可以继续使用，如果你想继续使用，就用 while，否则推荐使

用 for。原因是 for 循环结束，该变量就从内存中消失，能够提高内存的使用效率。

**场景区别：**

for 循环适合针对一个范围判断进行操作

while 循环适合判断次数不明确操作

**循环嵌套**

循环嵌套，顾名思义就是循环嵌套循环直接举两个小栗子让大家理解一下

栗子一：

```
/**
    输出如下图形：
        *
        **
        ***
        ****
        *****
*/
class Demo
{
    public static void main(String[] args){
        for(int x = 1; x <= 5; x++){
            for(int y = 1; y <= x; y++){
                System.out.print("*" );
            }
            System.out.println();
        }
    }
}
```

复制代码

栗子二：

```
/**
* 需求：打印出 99 乘法表
**/
class Demo {
        public static void main(String[] args) {
                for (int x = 1; x <= 9; x++) {
                        for (int y = 1; y <= x; y++) {
                                System.out.print(y + "*" + x + "=" + (x * y) + "\t");
                        }
                        System.out.println();
                }
        }
}
```

复制代码

那如果我们想在某个循环到某一步的时候就结束该怎么办呢？Java 就提供了 **break，continue 和 return** 来实现循环语句的跳转和中断。

**break**

使用场景：只能在 switch 和循环中，既可以跳出单层循环又可以跳出多层循环

**continue**

使用场景 ：只能在循环中，只能退出本次循环

**return**

return 关键字不是为了跳转出循环体，更常用的功能是结束一个方法，也就是退出一个方法。跳转到上层调用的方法

## 方法概述和格式说明

### 使用方法的好处

提高代码的复用性

### 什么是方法

完成特定功能的代码块。

### 格式

```
修饰符 返回值类型 方法名(参数类型 参数名1,参数类型 参数名2...) {
            方法体语句;
            return 返回值;
        }
```

复制代码

### 格式说明

修饰符：public protected private  static

返回值类型：就是功能结果的数据类型。

方法名：符合命名规则即可。方便我们的调用。

参数：

实际参数：就是实际参与运算的。

形式参数；就是方法定义上的，用于接收实际参数的。

参数类型：就是参数的数据类型

参数名：就是变量名

方法体语句：就是完成功能的代码。

return：结束方法的。

返回值：就是功能的结果，由 return 带给调用者。

写方法之前我们明确返回值类型和参数列表

例如：

```
/**
        *
        **
        ***
        ....
        打印N行星星
*/
class Demo {
        public static void main(String[] args) {
                printStart(6);
        }

        private static void printStart(int b) {
                for (int x = 1; x <= b; x++) {
                        for (int y = 1; y <= x; y++) {
                                System.out.print("*");
                        }
                        System.out.println();
                }
        }
}
```

复制代码

**方法的注意事项**

a:方法不调用不执行

b:方法与方法是平级关系，不能嵌套定义

c:方法定义的时候参数之间用逗号隔开

d:方法调用的时候不用在传递数据类型

e:如果方法有明确的返回值，一定要有 return 带回一个值

**方法重载概述和基本使用**

方法重载概述

在同一个类中，方法名相同，参数列表不同。与返回值类型无关。

参数列表不同：

A:参数个数不同

B:参数类型不同

C:参数的顺序不同(算重载,但是在开发中不用)

比较两个数据是否相等。

案例：

```
/**
* 
* 重载方法演示
*/
class Demo {
	public static void main(String[] args) {
		print(1);
		print(1, 2);
		print("1", "2");
	}

	public static void print(int a) {
		System.out.println("a=" + a);
	}

	public static void print(int a, int b) {
		System.out.println("(a+b)=" + (a + b));
```

```
18.         }
19.
20.         public static void print(String a, String b) {
21.                 System.out.println("(a+b)=" + (a + b));
22.         }
23. }
```

复制代码