

一、判断

if语句

If语句

格式：

一种情况的判断

```
if (关系表达式) {  
    语句体;  
}
```

格式：

两种情况的判断

```
if (关系表达式) {  
    语句体1;  
} else {  
    语句体2;  
}
```

If语句的第三种格式

格式：

```
if (关系表达式1) {  
    语句体1;  
} else if (关系表达式2) {  
    语句体2;  
}  
...  
else {  
    语句体 n + 1;  
}
```

执行流程：

- I. 首先计算关系表达式1的值
- II. 如果为true就执行语句体1；如果为false就计算关系表达式2的值
- III. 如果为true就执行语句体2；如果为false就计算关系表达式3的值
- IV.
- V. 如果所以关系表达式结果都为false，就执行语句体n+1。

switch语句

```
/**
```

```
* 如果说，要对一个范围进行判断，则需要使用if
```

```
* 如果对，有限个结果，一一列举出来，任选其一，则需要使用switch
```

```
*/
```

switch语句格式说明

```
switch(表达式) {  
    case 值1:  
        语句体1;  
        break;  
    case 值2:  
        语句体2;  
        break;  
    ...  
    default:  
        语句体n+1;  
        break;  
}
```

格式说明:

- I. 表达式: (**将要匹配的值**) 取值为byte、short、int、char。
JDK5以后可以是枚举, JDK7以后可以是String。
- II. case: 后面跟的是要和表达式进行比较的值 (**被匹配的值**)。
- III. break: 表示中断, 结束的意思, 用来结束switch语句。
- IV. default: 表示所有情况都不匹配的时候, 就执行该处的内容, 和if语句的else相似。
- V. case后面的值只能是字面量, 不能是变量
- VI. case给出的值不允许重复

• 注意

◦ 1.switch正常执行流程

- 执行流程
- 首先还是会拿着小括号中表达式的值跟下面每一个case进行匹配.如果匹配上了,就会执行对应的语句体,如果此时发现了break,那么结束整个switch语句:如果没有发现break,那么程序会继续执行下一个case的语句体,一直遇到break.或者右大括号为止.

使用场景:

如果多个case的语句体重复了,那么我们考虑利用case穿透去简化代码

▪

◦ 2.switch新特性 (JDK12)

```
/**  
 * JDK12的新特性  
 * switch case 的简化书写方式  
 *  
 * break; 都不用写了  
 */  
int a = 1;  
switch (a) {  
    case 1 -> {  
        a++;  
        System.out.println(a);  
    }  
    case 2 -> {  
        System.out.println("a为2");  
    }  
}
```

```
/**  
 * 如果case 中的语句体 只有一句话, 则可以省略大括号  
 * 大括号可以省略, break也可以省略  
 */  
int b = 10;  
switch (b) {  
    case 10 -> System.out.println("b为10");  
    case 20 -> System.out.println("b为20");  
    case 30 -> System.out.println("b为30");  
}
```

◦ 3.case穿透

- 就是语句体中没有写break导致的:

- 应用场景

```
/**
 * 键盘录入星期数, 输出工作日, 还是休息日
 * case 穿透的应用场景
 */
System.out.println("请输入星期数");
int week = new Scanner(System.in).nextInt();
switch (week) {
    case 1, 2, 3, 4, 5:
        System.out.println("工作日");
        break;
    case 6, 7:
        System.out.println("休息日");
        break;
    default:
        System.out.println("你输入的星期数不存在!");
}

/**
 * 拉姆达表达式简化写法
 */
System.out.println("请输入星期数");
int week = new Scanner(System.in).nextInt();
switch (week) {
    case 1, 2, 3, 4, 5 -> System.out.println("工作日");
    case 6, 7 -> System.out.println("休息日");
    default -> System.out.println("你输入的星期数不存在!");
}
```

• 4. default的位置和省略

- 位置:default 不一定是写在最下面的,我们可以写在任意位置.只不过习惯会写在最下面
- 省略:default可以省略,语法不会有问题,但是不建议省略.

二、循环

- 在实际开发中, 我们需要重复的执行某段代码, 会选择循环来实现

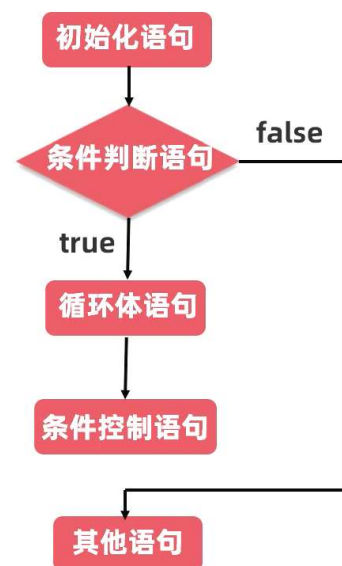
1.for循环

for循环

格式:
`for (int i = 1; i <= 10; i++) {
 System.out.println("HelloWorld");
}`

执行流程:

- ① 执行初始化语句
- ② 执行条件判断语句, 看其结果是true还是false
如果是false, 循环结束
如果是true, 执行循环体语句
- ③ 执行条件控制语句
- ④ 回到②继续执行条件判断语句



```

/**
 * 统计满足条件的数字
 * 键盘录入两个数字，表示一个范围
 * 统计这个范围中既能被3整除又能被5整除数字有多少个
 *
 * 统计的思想 用来计数
 */

System.out.println("请输入两个数字");
int i = new Scanner(System.in).nextInt();
int n = new Scanner(System.in).nextInt();
//统计个数
int sum=0;
for (int j = i; j <= n; j++) {
    if(j%3==0&&j%5==0){
        System.out.println(j);
        sum++;
    }
}
System.out.println("在"+i+"到"+n+"范围内，既能被3整除又能被5整除数字总共有"+sum+"个");

```

2.while循环

while循环的执行流程

格式：

```

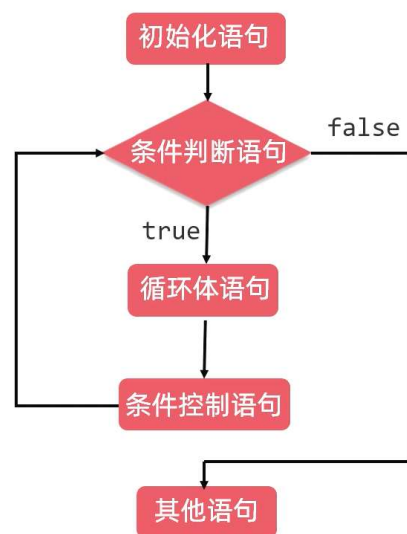
初始化语句；
while(条件判断语句){
    循环体语句；
    条件控制语句；
}
循环下面的其他语句

```

初始化语句只执行一次

判断语句为true，循环继续

判断语句为false，循环结束



3.for 和while的区别

for和while的对比

- 相同点：运行规则都是一样的

for 和 while 的区别：**使用习惯上的区别**

- for循环中：知道循环次数或者循环的范围
- while循环：不知道循环的次数和范围，只知道循环的结束条件。

```
int i = 0; 这个变量i可以提出去使用
for( ; i < 4; i++) {
    System.out.println(i);
}
```

```
int i = 0;
while (i < 4) {
    i++;
}
System.out.println(i);
```

4.无限循环

- 循环一直停不下来
-

```
for (;;) {
    System.out.println("学习");
}
```

```
while(true) { 常用
    System.out.println("学习");
}
```

```
do {
    System.out.println("学习");
} while(true);
```

- 注意事项

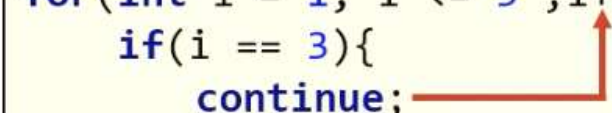
- 在无限循环的下面，不能再写其他代码了，因为循环永远停不下来，那么下面的代码永远执行不到

5.跳转控制语句

- 1.continue:跳过本次循环,继续执行下次循环.

◦

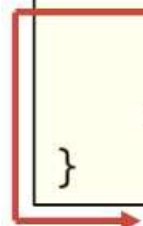
```
for(int i = 1; i <= 5 ;i++){  
    if(i == 3){  
        continue;  
    }  
    System.out.println("吃第" + i + "个包子");  
}
```



- 2. break:结束整个循环.

◦

```
for(int i = 1 ; i <= 5 ; i++){  
    if(i == 3){  
        break;  
    }  
    System.out.println("吃第" + i + "个包子");  
}
```



© 版权声明

版权声明

1. 本网站名称：ΛMΛ
 2. ΛMΛ提供的资源仅供您个人用于非商业性目的。
 3. 本站文章部分内容可能来源于网络，仅供大家学习与参考，如有侵权，请联系我进行删除处理。
 4. 本站一切资源不代表本站立场，并不代表本站赞同其观点和对其真实性负责。
 5. 本站一律禁止以任何方式发布或转载任何违法的相关信息，访客发现请举报
 6. 本站资源大多存储在云盘，如发现链接失效，请联系我，我会第一时间更新。
 7. 本站强烈打击盗版/破解等有损他人权益和违法作为，请支持正版！
-

