

选择结构 switch 语句的格式及其解释(掌握)

- A:switch 语句的格式

▪

```
switch(表达式) {  
    case 值 1:  //case: 如果  
        语句体 1;  
        break;  
    case 值 2:  
        语句体 2;  
        break;  
    ...  
    default: //缺省等同于 else  
        语句体 n+1;  
        break;  
}
```

- B:switch 语句的格式解释

- C:面试题

- byte 可以作为 switch 的表达式吗?
- long 可以作为 switch 的表达式吗?
- String 可以作为 switch 的表达式吗?

- C:执行流程

- 先计算表达式的值
- 然后和 case 后面的匹配, 如果有就执行对应的语句, 否则执行 default 控制的语句

```
public class Demo7 {  
  
    public static void main(String[] args) {  
  
        int floor=7;  
  
        switch (floor){
```

```
        case 1:

            System.out.println("到了第一层");

            break; // 中止

        case 2:

            System.out.println("到了第二层");

            break;

        case 3:

            System.out.println("到了第三层");

            break;

        case 4:

            System.out.println("到了第四层");

            break;

        default: // 处理以上都不满足的情况

            System.out.println("没有你想去的楼层");

            break;

    }

}

}
```

```
public class Demo7 {

    public static void main(String[] args) {

        int floor=3;

        switch (floor){
```

```
case 1:

    System.out.println("到了第一层");

    break; // 中止, 如果没有break, 那么会发生穿透的现象, 这个break
一定不能丢

case 2:

    System.out.println("到了第二层");

    break;

case 3:

    System.out.println("到了第三层");

    break;

case 4:

    System.out.println("到了第四层");

    break;

default: // 处理以上都不满足的情况, 一般情况下default 放在最后
写, default 的执行顺序是所有的 case 之后

    System.out.println("没有你想去的楼层");

    break; // 如果 default 在最后, 这个break 可以去掉

}

}

}
```

● 选择结构 switch 语句的练习 (掌握)

- A: 案例演示 1

整数(给定一个值, 输出对应星期几)

```
import java.util.Scanner;

public class Demo7 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("请输入一个星期数: ");

        int week = sc.nextInt();

        switch (week){

            case 1:

                System.out.println("周一");

                break;

            case 2:

                System.out.println("周二");

                break;

            case 3:

                System.out.println("周三");

                break;

            case 4:

                System.out.println("周四");

                break;

            case 5:
```

```
        System.out.println("周五");

        break;

    case 6:

        System.out.println("周六");

        break;

    case 7:

        System.out.println("周日");

        break;

    default:

        System.out.println("输入有误，请重新输入！");

    }

}

}
```

```
import java.util.Scanner;

public class Demo7 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("请输入一个月份数：");

        int week = sc.nextInt();

        switch (week){ //switch 的括号内可以接受的数据类型：
                        byte\short\int\long\char\String\枚举
        }
```

```
        case 3:

        case 4:

        case 5:

            System.out.println("春暖花开");

            break;

        case 6:

        case 7:

        case 8:

            System.out.println("夏日清爽");

            break;

        case 9:

        case 10:

        case 11:

            System.out.println("秋高气爽");

            break;

        case 12:

        case 1:

        case 2:

            System.out.println("冻死个人");

        default:

            System.out.println("输入有误, 请重新输入! ");

    }
```

<pre>} }</pre>
switch 使用予非范围判断的控制条件中 如果是在一个范围内，优先选用 if_ else if_ else

● 选择结构 switch 语句的注意事项(掌握)

▪ A: 案例演示 2

- a: case 后面只能是常量，不能是变量，而且，多个 case 后面的值不能出现相同的

- b: default 可以省略吗？

- 可以省略，但是不建议，因为它的作用是对不正确的情况给出提示。

- 特殊情况：

- case 就可以把值固定。

- A, B, C, D

- c: break 可以省略吗？

- 最后一个可以省略，其他最好不要省略

- 会出现一个现象：case 穿透。

- 最终我们建议不要省略

- d: default 一定要在最后吗？

- 不是，可以在任意位置。但是建议在最后。

- e: switch 语句的结束条件

- a: 遇到 break 就结束了

- b: 执行到 switch 的右大括号就结束了

- 举例：皇帝翻牌子

```
import java.util.Scanner;

public class Demo7 {

    public static void main(String[] args) {

        String feizi = "李嬷嬷";

        switch (feizi){

            case "张贵妃":

                System.out.println("她老爸是将军，好好努力！");

                break;

            case "李嬷嬷":

                System.out.println("李嬷嬷比较凶，会扎针，小心点");

                break;

            case "陈皇后":

                System.out.println("琵琶弹的好，温柔体贴");

                break;

            case "张爱妃":

                System.out.println("做的糕点好吃");

                break;

            default:

                System.out.println("没有抽到，太高兴了，终于可以休息一天了");

        }

    }

}
```



```
}  
  
}  
  
•
```

● 选择结构 switch 语句练习 (掌握)

▪ A:看程序写结果:

```
public class Demo{  
    public static void main(String[]args){  
  
        int x =2;  
        int y =3;  
        switch(x){  
            default:  
                y++;  
            break;  
            case 3:  
                y++;  
            case 4:  
                y++;  
        }  
        System.out.println("y="+y);  
    }  
}
```

B:看程序写结果:

```
public class Demo{  
    public static void main(String[]args){  
  
        int x =2;  
        int y =3;  
        switch(x){  
            default:  
                y++;  
            case 3:  
                y++;  
            case 4:  
                y++;  
        }  
        System.out.println("y="+y);  
    }  
}
```

```
}
}
```

●选择结构 if 语句和 switch 语句的区别(掌握)

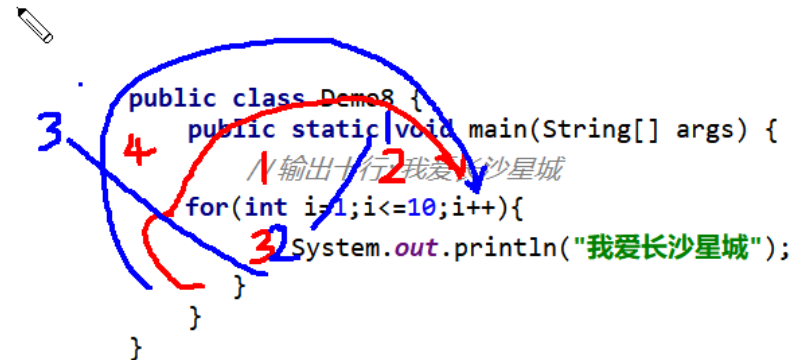
- A:总结 switch 语句和 if 语句的各自使用场景
 - switch 建议判断固定值的时候用
 - if 建议判断区间或范围的时候用
- B:案例演示
 - 分别用 switch 语句和 if 语句实现下列需求:
 - 键盘录入月份, 输出对应的季节

Java 语言基础(循环结构概述和 for 语句的格式及其使用)

- 需求: 要求控制台输出 10 行“我爱长沙星城”

```
public class Demo8 {
    public static void main(String[] args) {
        //输出十行我爱长沙星城
        System.out.println("我爱长沙星城");
        System.out.println("我爱长沙星城");
        System.out.println("我爱长沙星城");
        System.out.println("我爱长沙星城");
        System.out.println("我爱长沙星城");
        System.out.println("我爱长沙星城");
        System.out.println("我爱长沙星城");
        System.out.println("我爱长沙星城");
        System.out.println("我爱长沙星城");
        //输出一万行
    }
}
```

```
public class Demo8 {
    public static void main(String[] args) {
        //输出十行: 我爱长沙星城
        for(int i=1;i<=10;i++){
            System.out.println("我爱长沙星城");
        }
    }
}
```



```

public class Demo8 {
    public static void main(String[] args) {
        // 输出十行: 我爱长沙星城
        for(int i=1; i<=10; i++){
            System.out.println("我爱长沙星城");
        }
    }
}

```

首次是按红色数字顺序运行，从第二次开始到结束一直按蓝色的数字顺序运行

A: 循环结构的分类

for, while, do...while

B: 循环结构 for 语句的格式:

```

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

```

```

public class Demo8 {
    public static void main(String[] args) {
        // 输出十行: 我爱长沙星城
        int i=1; // 局部变量
        for(int i=1; i<=10; i++){ // 在 for 循环里的变量 i 的有效范围只在 for 循环体中有效, 当 for 循环
            System.out.println("我爱长沙星城");
        }
        // System.out.println("i="+i); // 出了 for 循环变量 i 无法使用
    }
}

```

C 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 for 继续。

D: 案例演示

在控制台输出 10 次"山上有座庙, 庙里有老和尚和小和尚, 老和尚给小和尚讲故事,

说"

```
public class Demo8 {  
    public static void main(String[] args) {  
        for(int i=0;i<10;i++){  
            System.out.println("山上有座庙，庙里有老和尚和小和尚，老和尚  
给小和尚讲故事，说");  
        }  
    }  
}
```

●循环结构 for 语句的练习之获取数据

A:案例演示

需求 1: 请在控制台输出数据 1-10

需求 2: 请在控制台输出数据 10-1

```
public class Demo8 {  
    public static void main(String[] args) {  
        //输出 1~10  
        for(int i=1;i<=10;i++){  
            System.out.print(i + " ");  
        }  
        System.out.println();//输出换行  
        //输出 10~1  
        for(int i=10;i>=1;i--){  
            System.out.print(i+" ");  
        }  
    }  
}
```

B:注意事项

判断条件语句无论简单还是复杂结果是 boolean 类型。

●循环结构 for 语句的练习之求和思想

A:案例演示

需求: 求出 1-10 之间数据之和

```
public class Demo8 {  
    public static void main(String[] args) {  
        //求 1~10 中所有数的和  
        int sum=0;//声明一个变量用来存储求和的结果，这个 sum 变量所在的位置要  
放在 for 循环之外  
  
        for(int i=1;i<=10;i++){
```

```

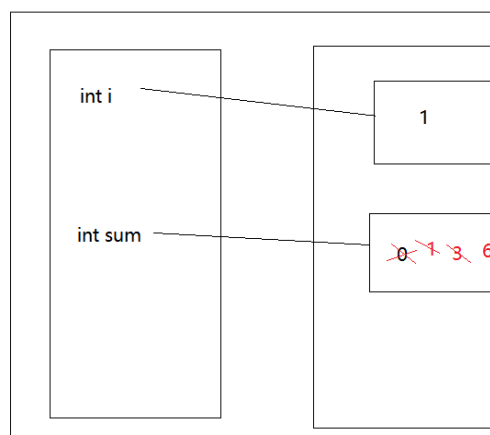
        sum += i;
    }
    System.out.println("求和的结果是: "+sum);
}
}

```

```

public class Demo8 {
    public static void main(String[] args) {
        // 求1~10中所有数的和
        int sum=0; // 声明一个变量用来存储求和
        for(int i=1;i<=10;i++){
            // System.out.print(i+" ");
            sum += i;
        }
        System.out.println("求和的结果是: "+sum);
    }
}

```



B:学生

需求: 求出 1-100 之间偶数和

练习: 求出 1-100 之间奇数和

```

public class Demo1 {
    public static void main(String[] args) {
        // 求出 1~100 之间的所有数的和
        // 申明接收求和结果的变量
        int sum=0;
        int sum2= 0; // 用来接收奇数的和
        for(int a=1;a<=100;a++){
            // 求偶数的和
            if(a % 2 == 0) { // 能被 2 整除的是偶数
                sum += a;
            }else{
                // 接收奇数的和
                sum2 += a;
            }
        }
        System.out.println("偶数的和: "+sum);
        System.out.println("奇数的和: "+sum2);
    }
}

```

●循环结构 for 语句的练习之水仙花

A:案例演示

需求：在控制台输出 100 ~ 999 所有的“水仙花数”

所谓的水仙花数是指一个三位数，其各位数字的立方和等于该数本身。

举例：153 就是一个水仙花数。

$$153 = 1*1*1 + 5*5*5 + 3*3*3 = 1 + 125 + 27 = 153$$

```
public class Demo2 {  
    public static void main(String[] args) {  
        for(int i=100;i<1000;i++){  
            //获取个十百每一位上的数字  
            int ge = i%10;  
            int shi = i/10%10;  
            int bai = i/100;  
  
            if(ge*ge*ge + shi*shi*shi+bai*bai*bai == i){  
                System.out.println(i);  
            }  
        }  
    }  
}
```

int 345

个位: $345 \% 10 = 5$

十位: $345 / 10 = 34 \% 10 = 4$

百位: $345 / 100 = 3$

●循环结构 for 语句的练习之统计思想

A:案例演示

需求：统计 100 ~ 1000 所有的“水仙花数”共有多少个

```
public class Demo2 {  
    public static void main(String[] args) {  
        // 申明一个变量来对水仙花进行计数，要写在循环外  
        int num = 0;  
  
        for(int i=100;i<1000;i++){  
            // 获取个十百每一位上的数字  
            int ge = i%10;  
            int shi = i/10%10;  
            int bai = i/100;  
  
            if(ge*ge*ge + shi*shi*shi+bai*bai*bai == i){  
                System.out.println(i);  
                num++;  
            }  
            ++num; // 满足水仙花的条件就对 num 作 num+1 运算  
        }  
        System.out.println("水仙花数是: "+num);  
    }  
}
```

●while 语句的格式和基本使用

A:循环结构 while 语句的格式:

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

B:执行流程:

- 1:执行初始化语句
- 2: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
- 3:执行循环体语句;
- 4:执行控制条件语句

C:案例演示

需求: 用 while 循环在控制台输出数据: 1-10

```
public class Demo1 {  
    public static void main(String[] args) {
```

```
//输出 1~10
int i =1;
while(i<=10){ //当i<=10
    System.out.print(i+" ");
    i++;//控制循环的条件
}
}
```

```
public class Demo1 {
    public static void main(String[] args) {
        //输出 10~1
        int i =10;
        while(i>=1){
            System.out.print(i+" ");
            i--;//控制循环的条件
        }
    }
}
```

●while 语句的练习

A:求和思想

求 1-100 之和

```
public class Demo1 {
    public static void main(String[] args) {
        int sum=0;
        int i =1;
        while(i<=100){
            sum +=i;
            i++;//控制循环的条件，最好放在最后一行
        }
        System.out.println(sum);
    }
}
```



```
public class Demo1 {
    public static void main(String[] args) {
        int sum=0;
        int i =1;
        while(i<=100){
            sum +=i;
            i++; //控制循环的条件, 最好放在最后一行
        }
        System.out.println(sum);
    }
}
```

(Diagram: A red arrow points from the condition `i<=100` to the increment `i++`, indicating that the condition is checked after the increment. A grey arrow points from the `while` loop header to the loop body.)

B:统计思想

统计 100~1000"水仙花数"共有多少个

```
public class Demo1 {
    public static void main(String[] args) {
        //声明一个记录个数的变量
        int num = 0;
        int n = 100;
        while(n<1000){
            int ge = n%10;
            int shi = n/10%10;
            int bai = n/100%10;
            if(ge*ge*ge+shi*shi*shi+bai*bai*bai == n){
                num++;
            }
            //控制循环的条件
            n++;
        }
        System.out.println("水仙花数是: "+num);

        System.out.println("-----for 循环也可以这么写, 了解即可, 开发中不推荐这么写-----");
        int num2 = 0;
        int b=100;
        for(;b<1000;){
            int ge = b%10;
            int shi = b/10%10;
            int bai = b/100%10;

```

```
        if (ge*ge*ge+shi*shi*shi+bai*bai*bai == b){
            num2++;
        }
        b++;
    }
    System.out.println(num2);
}
}
```

● 循环结构 do...while 语句的格式和基本使用

A: 循环结构 do...while 语句的格式:

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

B: 执行流程:

- 1: 执行初始化语句
- 2: 执行循环体语句;
- 3: 执行控制条件语句
- 4: 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: 案例演示

需求: 请在控制台输出数据 1-10

```
public class Demo1 {
    public static void main(String[] args) {
        int n=19;
        do{ //先干了再说 (先斩后奏)
            System.out.println(n);
            n++;
        }while (n<=10);
    }
}

/*
do_while 无论是否满足条件, 都会运行一次 (将在外, 军命有所不为)
*/
```

特点：
无论是否满足条件都会输出一次结果

●循环结构三种循环语句的区别

案例演示

A:案例演示

三种循环语句的区别：

do...while 循环至少执行一次循环体。

而 for,while 循环必须先判断条件是否成立，然后决定是否执行循环体语句。

for 循环和 while 循环的区别：

A:如果你想在循环结束后，继续使用控制条件的那个变量，用 while 循环，否则用 for 循环。
不知道用谁就用 for 循环。因为变量及早的从内存中消失，可以提高内存的使用效率。

```
import java.util.Scanner;

public class Demo1 {
    public static void main(String[] args) {
        for(int i=1;i<=10;i++){
            System.out.print(i+" ");
        }
        // System.out.println(i); // 在 for 循环之后，循环中的 i 会被及时释
        // 放内存
        System.out.println();

        System.out.println("-----");
        int num=1;
        while(num<=10){
            System.out.print(num+",");
            num++;
        }
        System.out.println();
        System.out.println(num); // while 循环用过的 num 是可以继续使用的，如果你有需要继续使用循环的初始化变量的时候，选择 while 循环
    }
}
```

●注意事项：死循环

A:一定要注意控制条件语句控制的那个变量的问题，不要弄丢了，否则就容易死循环。

B:两种最简单的死循环格式

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

```
public class Demo1 {
    public static void main(String[] args) {
        if(true) {
            System.out.print(1);
        }
        //      for(;;){
        //          System.out.println(2);
        //      }

        //      System.out.println("死循环之后的程序没法到达");//开发中一定要避免
        //      避免出现死循环。程序员开发的禁忌
    }
}
```

```
import java.util.Scanner;

public class Demo1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();//用户在操作的时候，输入的信息，构成了死循环
        if(a!=1000) {    //解决：避免用户输入 1000
            while (a == 1000) {
                System.out.print(1);
            }
        }
        //      for(;;){
        //          System.out.println(2);
        //      }

        //      System.out.println("死循环之后的程序没法到达");//开发中一定要避免
        //      避免出现死循环。程序员开发的禁忌
    }
}
```

●循环结构循环嵌套输出 4 行 5 列的星星

A:案例演示

需求：请输出 4 行 5 列图案

B:结论:

外循环控制行数，内循环控制列数

```
public class Demo1 {
    public static void main(String[] args) {
        for(int j=0;j<5;j++) {    //外层循环，控制重复执行下面的五次

            //是输出一行五个*，把这里看作一个整体
            for (int i = 0; i < 5; i++) {
                System.out.print("*");
            }
            System.out.println();//输出换行
        }
    }
}
```

●循环结构循环嵌套输出正三角形

需求：请输出下列的形状

*
**

方法一:

```
public class Demo1 {
    public static void main(String[] args) {
        int n=1;//控制每行输出的个数
        for(int j=0;j<5;j++) {    //外层循环，控制重复执行下面的五次

            //是输出一行五个*，把这里看作一个整体
            for (int i = 0; i < n; i++) {
                System.out.print("*");
            }

            System.out.println();//输出换行
        }
    }
}
```

```
n++; //放在外层循环中

    }
}
}
```

方法二

```
public class Demo1 {
    public static void main(String[] args) {
        for(int j=1;j<=5;j++) { //外层循环，控制重复执行下面的五次

            //是输出一行五个*，把这里看作一个整体
            for (int i = 0; i < j; i++) { //j 整合记录的行数，每行*的个数=行
数
                System.out.print("*");
            }

            System.out.println();//输出换行
        }
    }
}
```

●循环结构九九乘法表

- A:案例演示
- 需求：在控制台输出九九乘法表。

```
public class Demo1 {
    public static void main(String[] args) {
        for(int j=1;j<=9;j++) { //外层循环，控制重复执行下面的五次

            //是输出一行五个*，把这里看作一个整体
            for (int i = 1; i <= j; i++) { //j 整合记录的行数，每行*的个数=
行数
                System.out.print(i+"*"+j+"="+i*j+" ");
            }

            System.out.println();//输出换行
        }
    }
}
```

- B:代码优化

- 注意:
- '\x' x 表示任意, \是转义符号,这种做法叫转移字符。
- '\t' tab 键的位置 (单引号, 双引号都可以)
- '\r' 回车
- '\n' 换行
- '\"'
- '\\'

```
public class Demo2 {  
    public static void main(String[] args) {  
        //      System.out.println("\\"); //在这里\是转义符, 它的后面接要被转义  
        //      的字符  
        //      System.out.print("你好");  
        //      System.out.print("\\r"); //回车, 用在linux (作服务器用) 系统中  
        //      使用  
        //      System.out.print("我很好");  
        //      System.out.println("\\n"); //回车, 支持windows 和mac 系统  
        for(int i=0;i<10;i++){  
            System.out.print(i);  
            System.out.print("\\n");//回车  
            //System.out.print("\\n\\r");//这样写可以支持windows、mac、  
Linux 等几乎所有的操作系统  
        }  
        System.out.println();//这也是换行  
        for(int i=1;i<20;i++){  
            System.out.print(i+"\\t"); // \\t 代表制表符等同于Tab 键  
        }  
    }  
}
```

windows 里面换行\n 都支持, 它来自于早期的键盘打字机
linux 系统换行支持\r mac 系统换行支持\n

●控制跳转语句 break 语句

- A:break 的使用场景
- 只能在 switch 和循环(loop)中

```
public class Demo2 {  
    public static void main(String[] args) {  
        for(int i=1;i<=10;i++){  
            if(i==5){  
                break;//中止当前循环  
            }  
            System.out.println(i);  
        }  
    }  
}
```

```

    }

    System.out.println("循环之后的程序");
}
}

```

break 只是中止循环（跳出循环体），但整个程序会继续运行

●控制跳转语句标号（了解）

- 标号:标记某个循环对其控制
- 标号组成规则:其实就是合法的标识符

```

public class Demo3 {
    public static void main(String[] args) {
        a:for(int i=1;i<=5;i++){ //外层循环,a 可以随便取名只要满足标识符的要求
            b:for (int j=1;j<=5;j++){ //内层循环, b 可以随便取名只要满足标识符的要求
                System.out.print("j="+j+"\t");
                if(j==2){
                    break b; //指定跳出的循环
                }
            }
            System.out.println();
            System.out.println("i="+i);
        }
    }
}

```

●控制跳转语句 continue 语句（了解）

- A:continue 的使用场景(中止本次循环，继续下一次循环)
- 只能在循环(loop)中

```

public class Demo3 {
    public static void main(String[] args) {
        for(int i=0;i<10;i++){
            if(i==4){
                continue; //跳出本次 (i=4) 循环, 继续下一次循环
            }
            System.out.print(i+"\t");
        }
    }
}

```



```
}
}
```

●控制调整语句练习

A:练习题（面试题）

要求：

- 1.在控制台输出 3 次："Java 班"
- 2.在控制台输出 7 次："Java 班"
- 3.在控制台输出 13 次："Java 班"

```
for(int x=1; x<=10; x++) {
    if(x%3==0) {
        //在这里写代码
    }
    System.out.println("Java 班");
}
```

输出 3 次

```
public class Demo3 {
    public static void main(String[] args) {
        for(int x=1; x<=10; x++) {
            if(x%3==0) {
                //在这里写代码
                System.out.println("java 班");
                break;//break 后面的代码没有机会运行
            }
            System.out.println("Java 班");
        }
    }
}
```

输出 7 次

```
public class Demo3 {
    public static void main(String[] args) {
        for(int x=1; x<=10; x++) {
            if(x%3==0) {
                //在这里写代码
                continue;//跳出本次循环，当i=3,6,9 的时候跳出循环，那么就输出的10 次中少了3 次
            }
            System.out.println("Java 班");
        }
    }
}
```

```
}
}
```

输出 13 次

```
public class Demo3 {
    public static void main(String[] args) {
        for(int x=1; x<=10; x++) {
            if(x%3==0) {
                //在这里写代码
                System.out.println("java 班");
            }
            System.out.println("Java 班");
        }
    }
}
```

●控制跳转语句 return 语句

A: return 的作用

返回

其实它的作用不是结束循环的，而是结束方法的。

案例演示

```
public class Demo3 {
    public static void main(String[] args) {
        for(int x=1; x<=10; x++) {
            System.out.println(x);
            if(x==5){
                return; //返回，结束的是整个所在的方法，return 后面不要有程序了，因为到达不了
            }
        }

        System.out.println("循环之后的程序"); //这句话不会被执行了
    }
}
```

return 和 break 以及 continue 的区别？

return 是结束它所在的方法

break 是跳出循环（for, while, switch）

continue 是中止本次循环继续下次循环(for, while, do_while)