

1. switch语句

1.1 分支语句switch语句

- 格式

```
1  switch (表达式) {  
2      case 1:  
3          语句体1;  
4          break;  
5      case 2:  
6          语句体2;  
7          break;  
8      ...  
9      default:  
10         语句体n+1;  
11         break;  
12 }
```

- 执行流程:

- 首先计算出表达式的值
- 其次, 和case依次比较, 一旦有对应的值, 就会执行相应的语句, 在执行的过程中, 遇到break就会结束。
- 最后, 如果所有的case都和表达式的值不匹配, 就会执行default语句体部分, 然后程序结束掉。

1.2 switch案例-减肥计划

- 需求: 键盘录入星期数, 显示今天的减肥活动

```
1  周一: 跑步  
2  周二: 游泳  
3  周三: 慢走  
4  周四: 动感单车  
5  周五: 拳击  
6  周六: 爬山  
7  周日: 好好吃一顿
```

- 示例代码:

```
1  public static void main(String[] args){  
2      // 1. 键盘录入星期数据, 使用变量接收  
3      Scanner sc = new Scanner(System.in);  
4      System.out.println("请输入");  
5      int week = sc.nextInt();  
6      // 2. 多情况判断, 采用switch语句实现  
7      switch(week){  
8          // 3. 在不同的case中, 输出对应的减肥计划
```

```

9         case 1:
10             System.out.println("跑步");
11             break;
12         case 2:
13             System.out.println("游泳");
14             break;
15         case 3:
16             System.out.println("慢走");
17             break;
18         case 4:
19             System.out.println("动感单车");
20             break;
21         case 5:
22             System.out.println("拳击");
23             break;
24         case 6:
25             System.out.println("爬山");
26             break;
27         case 7:
28             System.out.println("好好吃一顿");
29             break;
30         default:
31             System.out.println("您的输入有误");
32             break;
33     }
34 }
35 }

```

1.3 switch语句case穿透

- 概述：如果switch语句中,case省略了break语句,就会开始case穿透
- 需求：键盘录入星期数，输出工作日、休息日 (1-5)工作日，(6-7)休息日
- 示例代码：

```

1  /*
2  case穿透是如何产生的?
3
4      如果switch语句中,case省略了break语句,就会开始case穿透.
5
6      现象:
7      当开始case穿透,后续的case就不会具有匹配效果,内部的语句都会执行
8      直到看见break,或者将整体switch语句执行完毕,才会结束.
9  */
10 public static void main(String[] args){
11     Scanner sc = new Scanner(System.in);
12     System.out.println("请输入星期数:");
13     int week = sc.nextInt();
14
15     switch(week){
16         case 1:
17         case 2:
18         case 3:

```

```

19         case 4:
20         case 5:
21             System.out.println("工作日");
22             break;
23         case 6:
24         case 7:
25             System.out.println("休息日");
26             break;
27         default:
28             System.out.println("您的输入有误");
29             break;
30     }
31 }
32 }

```

2. for循环

2.1 循环语句-for循环

- 循环：

循环语句可以在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，当反复执行这个循环体时，需要在合适的时候把循环判断条件修改为false，从而结束循环，否则循环将一直执行下去，形成死循环。

- for循环格式：

```

1 for (初始化语句;条件判断语句;条件控制语句) {
2     循环体语句;
3 }

```

- 格式解释：

- 初始化语句：用于表示循环开启时的起始状态，简单说就是循环开始的时候什么样
- 条件判断语句：用于表示循环反复执行的条件，简单说就是判断循环是否能一直执行下去
- 循环体语句：用于表示循环反复执行的内容，简单说就是循环反复执行的事情
- 条件控制语句：用于表示循环执行中每次变化的内容，简单说就是控制循环是否能执行下去

- 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，继续执行

③执行循环体语句

④执行条件控制语句

⑤回到②继续

2.2 for循环案例-输出数据1-5和5-1

- 需求：在控制台输出1-5和5-1的数据
- 示例代码：

```
1 public class ForTest01 {
2     public static void main(String[] args) {
3         //需求：输出数据1-5
4         for(int i=1; i<=5; i++) {
5             System.out.println(i);
6         }
7         System.out.println("-----");
8         //需求：输出数据5-1
9         for(int i=5; i>=1; i--) {
10            System.out.println(i);
11        }
12    }
13 }
```

2.3 for循环案例-求1-5数据和

- 需求：求1-5之间的数据和，并把求和结果在控制台输出
- 示例代码：

```
1 public class ForTest02 {
2     public static void main(String[] args) {
3         //求和的最终结果必须保存起来，需要定义一个变量，用于保存求和的结果，初始值为0
4         int sum = 0;
5         //从1开始到5结束的数据，使用循环结构完成
6         for(int i=1; i<=5; i++) {
7             //将反复进行的事情写入循环结构内部
8             // 此处反复进行的事情是将数据 i 加到用于保存最终求和的变量 sum 中
9             sum += i;
10            /*
11                sum += i;    sum = sum + i;
12                第一次: sum = sum + i = 0 + 1 = 1;
13                第二次: sum = sum + i = 1 + 2 = 3;
14                第三次: sum = sum + i = 3 + 3 = 6;
15                第四次: sum = sum + i = 6 + 4 = 10;
16                第五次: sum = sum + i = 10 + 5 = 15;
17            */
18        }
19        //当循环执行完毕时，将最终数据打印出来
20        System.out.println("1-5之间的数据和是: " + sum);
21    }
22 }
```

- 本题要点：
 - 今后遇到的需求中，如果带有求和二字，请立即联想到求和变量
 - 求和变量的定义位置，必须在循环外部，如果在循环内部则计算出的数据将是错误的

2.4 for循环案例-求1-100偶数和

- 需求：求1-100之间的偶数和，并把求和结果在控制台输出 }
- 示例代码：

```
1 public class ForTest03 {
2     public static void main(String[] args) {
3         //求和的最终结果必须保存起来，需要定义一个变量，用于保存求和的结果，初始值为0
4         int sum = 0;
5         //对1-100的数据求和与1-5的数据求和几乎完全一样，仅仅是结束条件不同
6         for(int i=1; i<=100; i++) {
7             //对1-100的偶数求和，需要对求和操作添加限制条件，判断是否是偶数
8             if(i%2 == 0) {
9                 sum += i;
10            }
11        }
12        //当循环执行完毕时，将最终数据打印出来
13        System.out.println("1-100之间的偶数和是：" + sum);
14    }
15 }
```

2.5 for循环案例-水仙花数

- 需求：在控制台输出所有的“水仙花数”
- 解释：什么是水仙花数？
 - 水仙花数，指的是一个三位数，个位、十位、百位的数字立方和等于原数
 - 例如 $153 \quad 3*3*3 + 5*5*5 + 1*1*1 = 153$
- 思路：
 1. 获取所有的三位数，准备进行筛选，最小的三位数为100，最大的三位数为999，使用for循环获取
 2. 获取每一个三位数的个位，十位，百位，做if语句判断是否是水仙花数
- 示例代码

```
1 public class ForTest04 {
2     public static void main(String[] args) {
3         //输出所有的水仙花数必然要使用到循环，遍历所有的三位数，三位数从100开始，到999结束
4         for(int i=100; i<1000; i++) {
5             //在计算之前获取三位数中每个位上的值
6             int ge = i%10;
7             int shi = i/10%10;
8             int bai = i/10/10%10;
9
10            //判定条件是将三位数中的每个数值取出来，计算立方和后与原始数字比较是否相等
11            if(ge*ge*ge + shi*shi*shi + bai*bai*bai == i) {
12                //输出满足条件的数字就是水仙花数
13                System.out.println(i);
14            }
15        }
16    }
17 }
```

2.6 for循环案例-每行打印2个水仙花数(统计)

- 需求：在控制台输出所有的“水仙花数”，要求每行打印2个
- 示例代码：

```
1 public class Demo6For {
2     /*
3         需求：在控制台输出所有的“水仙花数”，要求每行打印2个
4
5         System.out.print (打印内容);    打印后不换行
6         System.out.println(打印内容);    打印后换行
7
8         分析：
9
10        1. 定义变量count，用于保存“打印过”的数量，初始值为0
11        2. 在判定和打印水仙花数的过程中，拼接空格，但不换行，并在打印后让count变量+1，记录打印
12        过的数量
13        3. 在每一次count变量+1后，判断是否到达了2的倍数，是的话，换行。
14
15        */
16        public static void main(String[] args){
17            // 1. 定义变量count，用于保存“打印过”的数量，初始值为0
18            int count = 0;
19            for(int i = 100; i <= 999; i++){
20                int ge = i % 10;
21                int shi = i / 10 % 10;
22                int bai = i / 10 / 10 % 10;
23
24                if( (ge*ge*ge + shi*shi*shi + bai*bai*bai) == i){
25                    // 2. 在判定和打印水仙花数的过程中，拼接空格，但不换行，并在打印后让count变量+1，
26                    记录打印过的数量
27                    System.out.print(i + " ");
28                    count++;
29                    // 3. 在每一次count变量+1后，判断是否到达了2的倍数，是的话，换行
30                    if(count % 2 == 0){
31                        System.out.println();
32                    }
33                }
34            }
35        }
36    }
```

- 本题要点：
 - 今后如果需求带有统计xxx，请先想到计数器变量
 - 计数器变量定义的位置，必须在循环外部

3. while循环

3.1 循环语句-while循环

- while循环完整格式：

```
1  初始化语句;
2  while (条件判断语句) {
3      循环体语句;
4      条件控制语句;
5  }
```

- while循环执行流程:

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, 继续执行

③执行循环体语句

④执行条件控制语句

⑤回到②继续

- 示例代码:

```
1  public class WhileDemo {
2      public static void main(String[] args) {
3          //需求: 在控制台输出5次"HelloWorld"
4          //for循环实现
5          for(int i=1; i<=5; i++) {
6              System.out.println("HelloWorld");
7          }
8          System.out.println("-----");
9          //while循环实现
10         int j = 1;
11         while(j<=5) {
12             System.out.println("HelloWorld");
13             j++;
14         }
15     }
16 }
```

3.2 while循环案例-珠穆朗玛峰

- 需求: 世界最高山峰是珠穆朗玛峰(8844.43米=8844430毫米), 假如我有一张足够大的纸, 它的厚度是0.1毫米。请问, 我折叠多少次, 可以折成珠穆朗玛峰的高度?
- 示例代码:

```
1  public class WhileTest {
2      public static void main(String[] args) {
3          //定义一个计数器, 初始值为0
4          int count = 0;
5          //定义纸张厚度
6          double paper = 0.1;
7          //定义珠穆朗玛峰的高度
```

```

8      int zf = 8844430;
9      //因为要反复折叠，所以要使用循环，但是不知道折叠多少次，这种情况下更适合使用while循环
10     //折叠的过程中当纸张厚度大于珠峰就停止了，因此继续执行的要求是纸张厚度小于珠峰高度
11     while(paper <= zf) {
12         //循环的执行过程中每次纸张折叠，纸张的厚度要加倍
13         paper *= 2;
14         //在循环中执行累加，对应折叠了多少次
15         count++;
16     }
17     //打印计数器的值
18     System.out.println("需要折叠: " + count + "次");
19 }
20 }

```

4. 循环细节

4.1 循环语句-dowhile循环

- 完整格式：

```

1  初始化语句;
2  do {
3      循环体语句;
4      条件控制语句;
5  }while(条件判断语句);

```

- 执行流程：
 - ① 执行初始化语句
 - ② 执行循环体语句
 - ③ 执行条件控制语句
 - ④ 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，继续执行

- ⑤ 回到②继续

- 示例代码：

```

1  public class DoWhileDemo {
2      public static void main(String[] args) {
3          //需求：在控制台输出5次"HelloWorld"
4          //for循环实现
5          for(int i=1; i<=5; i++) {
6              System.out.println("HelloWorld");
7          }
8          System.out.println("-----");
9          //do...while循环实现
10         int j = 1;
11         do {

```



```

12         System.out.println("HelloWorld");
13         j++;
14     }while(j<=5);
15 }
16 }

```

4.2 三种循环的区别

- 三种循环的区别
 - for循环和while循环先判断条件是否成立，然后决定是否执行循环体（先判断后执行）
 - do...while循环先执行一次循环体，然后判断条件是否成立，是否继续执行循环体（先执行后判断）
- for循环和while的区别
 - 条件控制语句所控制的自增变量，因为归属for循环的语法结构中，在for循环结束后，就不能再次被访问到了
 - 条件控制语句所控制的自增变量，对于while循环来说不归属其语法结构中，在while循环结束后，该变量还可以继续使用
- 死循环（无限循环）的三种格式
 1. for(;;){}
 2. while(true){}
 3. do {} while(true);

4.3 死循环

- 死循环格式

```

1  for死循环格式 :
2  for(;;){
3
4  }
5
6  while死循环格式 :
7
8  while(true){
9
10 }
11
12 do..while死循环格式 :
13
14 do{
15
16 }while(true);

```

- 死循环案例

```

1  /*
2      问题：死循环有应用场景吗？
3
4      例如：键盘录入一个1-100之间的整数

```

```

5
6          顾虑：键盘录入是用户操作的，用户就可能会出现一些误操作的现象
7
8  */
9  public static void main(String[] args) {
10      /*
11          for(;;){
12              System.out.println("我停不下来了~");
13          }
14          */
15
16      /*
17          while(true){
18              System.out.println("我停不下来了~");
19          }
20          */
21
22      do{
23          System.out.println("我停不下来了~");
24      }while(true);
25
26      System.out.println("看看我能被执行吗?~");    // 无法访问的语句
27  }
28  }

```

4.4 跳转控制语句

- 跳转控制语句（break）
 - 跳出循环，结束循环
- 跳转控制语句（continue）
 - 跳过本次循环，继续下次循环
- 注意：continue只能在循环中进行使用！

```

1  public class Demo1Continue {
2      /*
3          continue : 跳过某次循环体内容的执行
4
5          注意：使用是基于条件控制，在循环内部使用。
6
7          需求：模拟电梯上行的过程 1-24层，4层不停。
8      */
9      public static void main(String[] args){
10          for(int i = 1; i <= 24; i++){
11              if(i == 4){
12                  continue;
13              }
14              System.out.println(i + "层到了~");
15          }

```

```

16     }
17
18 }

```

```

1  public class Demo2Break {
2      /*
3          break : 终止循环体内容的执行
4          注意: 使用是基于条件控制的
5              break语句只能在循环和switch中进行使用.
6
7          需求: 模拟20岁工作到80岁, 60岁退休.
8      */
9      public static void main(String[] args){
10         for(int i = 20; i <= 80; i++){
11             if(i == 60){
12                 break;        // 结束整个循环
13             }
14             System.out.println(i + "岁正在上班");
15         }
16     }
17
18 }

```

```

1  import java.util.Scanner;
2
3  public class Test {
4      /*
5          需求: 程序运行后, 用户可多次查询星期对应的减肥计划, 直到输入0, 程序结束
6
7          步骤:
8
9              1. 不明确用户操作几次, 使用死循环包裹业务逻辑
10             2. 匹配到0的时候, 使用break结束循环死循环
11
12      */
13      public static void main (String[] args){
14
15          lo:while(true){
16              System.out.println("请输入您要查看的星期数:");
17              System.out.println("(如无需继续查看,请输入0退出程序)");
18
19              // 1. 键盘录入星期数据, 使用变量接收
20              Scanner sc = new Scanner(System.in);
21              int week = sc.nextInt();
22              // 2. 多情况判断, 采用switch语句实现
23              switch(week){
24                  // 3. 在不同的case中, 输出对应的减肥计划
25                  case 0:
26                      System.out.println("感谢您的使用");
27                      break lo;
28
29                  case 1:

```

```
29         System.out.println("跑步");
30         break;
31     case 2:
32         System.out.println("游泳");
33         break;
34     case 3:
35         System.out.println("慢走");
36         break;
37     case 4:
38         System.out.println("动感单车");
39         break;
40     case 5:
41         System.out.println("拳击");
42         break;
43     case 6:
44         System.out.println("爬山");
45         break;
46     case 7:
47         System.out.println("好好吃一顿");
48         break;
49     default:
50         System.out.println("您的输入有误");
51         break;
52     }
53 }
54
55
56 }
57 }
```

5. Random

5.1 Random产生随机数（掌握）

- 概述：
 - Random类似Scanner，也是Java提供好的API，内部提供了产生随机数的功能
 - API后续课程详细讲解，现在可以简单理解为Java已经写好的代码
- 使用步骤：
 1. 导入包

```
import java.util.Random;
```
 2. 创建对象

```
Random r = new Random();
```
 3. 产生随机数

```
int num = r.nextInt(10);
```

解释：10代表的是一个范围，如果括号写10，产生的随机数就是0-9，括号写20，参数的随机数则是0-19

- 示例代码：

```
1  import java.util.Random;
2
3  public class Demo1Random {
4      /*
5          Random : 产生随机数
6
7          1. 导包    : import java.util.Random;
8                     导包的动作必须出现在类定义的上边
9
10         2. 创建对象 : Random r = new Random();
11                    上面这个格式里面，r 是变量名，可以变，其他的都不允许变
12
13         3. 获取随机数 : int number = r.nextInt(10); //获取数据的范围：[0,10) 包括0,不包括10
14                    上面这个格式里面，number是变量名，可以变，数字10可以变。其他的都不允许变
15
16         需求：产生随机数1-10之间的
17     */
18     public static void main(String[] args){
19         // 2. 创建对象
20         Random r = new Random();
21
22         for(int i = 1; i <= 10; i++){
23             // 3. 获取随机数
24             int num = r.nextInt(10) + 1;          // 1-10
25             System.out.println(num);
26         }
27
28
29
30     }
31 }
```

5.3 Random练习-猜数字（应用）

- 需求：

程序自动生成一个1-100之间的数字，使用程序实现猜出这个数字是多少？

当猜错的时候根据不同情况给出相应的提示

- A. 如果猜的数字比真实数字大，提示你猜的数据大了
- B. 如果猜的数字比真实数字小，提示你猜的数据小了
- C. 如果猜的数字与真实数字相等，提示恭喜你猜中了

- 示例代码：

```
1  import java.util.Scanner;
2  import java.util.Random;
```

```

3
4 public class Test {
5     /*
6         需求：程序自动生成一个1-100之间的数字，使用程序实现猜出这个数字是多少？
7         当猜错的时候根据不同情况给出相应的提示
8         如果猜的数字比真实数字大，提示你猜的数据大了
9         如果猜的数字比真实数字小，提示你猜的数据小了
10        如果猜的数字与真实数字相等，提示恭喜你猜中了
11
12        1. 准备Random和Scanner对象，分别用于产生随机数和键盘录入
13        2. 使用Random产生一个1-100之间的数，作为要猜的数
14        3. 键盘录入用户猜的数据
15        4. 使用录入的数据(用户猜的数据)和随机数(要猜的数据)进行比较，并给出提示
16
17        5. 以上内容需要多次进行，但无法预估用户输入几次可以猜测正确，使用while(true)死循环包裹
18        6. 猜对之后，break结束.
19
20    */
21    public static void main(String[] args){
22        // 1. 准备Random和Scanner对象，分别用于产生随机数和键盘录入
23        Random r = new Random();
24        Scanner sc = new Scanner(System.in);
25        // 2. 使用Random产生一个1-100之间的数，作为要猜的数
26        int randomNum = r.nextInt(100) + 1;
27
28        // 5. 以上内容需要多次进行，但无法预估用户输入几次可以猜测正确，使用while(true)死循环包裹
29        while(true){
30            // 3. 键盘录入用户猜的数据
31            System.out.println("请输入您猜的数据:");
32            int num = sc.nextInt();
33            // 4. 使用录入的数据(用户猜的数据)和随机数(要猜的数据)进行比较，并给出提示
34            if(num > randomNum){
35                System.out.println("猜大了");
36            }else if(num < randomNum){
37                System.out.println("猜小了");
38            }else{
39                // 6. 猜对之后，break结束.
40                System.out.println("恭喜,猜中了");
41                break;
42            }
43        }
44
45        System.out.println("感谢您的使用");
46
47    }
48 }

```