选择结构与分支结构

Author：zhangzhang

Version： 1.0.0

一、引言

1.1 选择结构

概念：根据已知条件进行逻辑判断，满足条件后执行相应操作。

二、 基本if选择结构

2.1 基本if选择结构

语法：
if(布尔表达式){
  //代码块
}

执行流程：

- 对布尔表达式进行判断。
- 结果为true，则先执行代码块，再执行后续代码。
- 结果为false，则跳过代码块，直接执行后续代码。

```
public class TestBasicIf{

  public static void main(String[] args){

    double score = 100.0;

    if( score == 100.0){
      System.out.println("恭喜，满分");
    }
```

```
    System.out.println("程序结束...");

  }
}
```

三、 if else选择结构【重点】

3.1 if else选择结构

> 语法：
> if(布尔表达式){
> //代码块1
> }else{
> //代码块2
> }

> 执行流程：
>
> - 对布尔表达式进行判断。
> - 结果为true，则先执行代码块1，再退出整个结构，执行后续代码。
> - 结果为false，则先执行代码块2，再退出整个结构，执行后续代码。

```
public class TestIfElse{

  public static void main(String[] args){

    double score = 100.0;

    if(score == 100.0){
      System.out.println("恭喜，满分");
    }else{
      System.out.println("抱歉，罚抄");
    }

    System.out.println("程序结束...");

    //程序猿 = 程序媛
  }
}
```

四、 多重if选择结构

4.1 多重if选择结构

> 语法：
> if(布尔表达式1){
> //代码块1
> }else if(布尔表达式2){
> //代码块2
> }else if(布尔表达式3){
> //代码块3
> }else{
> //代码块4
> }

> 执行流程：
>
> - 表达式1为true，则执行代码块1，再退出整个结构。
> - 表达式2为true，则执行代码块2，再退出整个结构。
> - 表达式3为true，则执行代码块3，再退出整个结构。
> - 以上均为false，则执行代码块4，再退出整个结构。

注意：相互排斥，有一个为true，其他均不再执行。

> 适用区间判断，但要保证条件顺序（从大到小、从小到大）。

```
public class TestMultipleIf{

  public static void main(String[] args){

    /*
      根据预算金额选购车辆

      预算 > 100万    奔驰S级
      预算 > 50万      宝马5系
```

```
      预算 > 10万      奥迪A4L
      预算 < 10万     捷安特自行车
    */

    int money = 110; //单位：万

    if(money > 100){
      System.out.println("奔驰S级");
    }else if(money > 50){
      System.out.println("宝马5系");
    }else if(money > 10){
      System.out.println("奥迪A4L");
    }else{
      System.out.println("捷安特自行车");
    }

    System.out.println("程序结束...");

  }
}
```

五、 嵌套if选择结构

5.1 嵌套if选择结构

语法：

```
if(外层表达式){
  if(内层表达式){
    //内层代码块1
  }else{
    //内层代码块2
  }
}else{
  //外层代码块
}
```

执行流程：

- 当外层条件满足时，再判断内层条件。

注意：[

- **一个选择结构中，可嵌套另一个选择结构。**
- **嵌套格式正确的情况下，支持任意组合。**

```
public class TestNestedIf{

  public static void main(String[] args){

    /*
      运动会百米赛跑
      用时10秒之内的人进入总决赛，否则淘汰
    */

    int timer = 9;

    char sex = '男';

    if(timer <= 10){//外层条件

      //进入总决赛
      if(sex == '男'){//内层条件
        System.out.println("男子组决赛");
      }else{
        System.out.println("女子组决赛");
      }

    }else{
      System.out.println("淘汰");
    }

  }
}
```

六、 switch分支结构【 重点 】

6.1 分支结构

语法：

```
switch(变量 | 表达式){
   case 值1:
      逻辑代码1;
   case 值2:
      逻辑代码2;
   case 值n:
      逻辑代码n;
      default:
        未满足时的逻辑代码;
}
```

可判断的类型：

- byte、short、int、char、String（JDK7+）

执行流程：

- 如果变量中的值等于值1，则执行逻辑代码1。
- 如果变量中的值等于值2，则执行逻辑代码2。
- 如果变量中的值等于值n，则执行逻辑代码n。
- 如果变量中的值没有匹配的case值时，执行default中的逻辑代码。

注意：当匹配的case执行后，不会自动退出整个结构，而是继续向下执行。

break关键字可在匹配的case执行后，跳出整个结构。

适用等值判断。

```
public class TestSwitch{

  public static void main(String[] args){

    /*
      每日食谱

      星期一：湘菜
      星期二：川菜
      星期三：粤菜
      星期四：浙菜
      星期五：川菜
      星期六：川菜
      星期日：徽菜

    */

    int weekDay = 8;

    switch( weekDay ){ //根据weekDay的值，找到匹配的case，并执行逻辑代码
      default:
        System.out.println("请输入1~7之间的整数!");
        break;
      case 1:
        System.out.println("湘菜");
        break;
      case 3:
        System.out.println("粤菜");
        break;
      case 4:
        System.out.println("浙菜");
        break;
      case 2:
      case 5:
      case 6:
        System.out.println("川菜");
        break;
      case 7:
        System.out.println("徽菜");
        break;
    }

    System.out.println("程序结束...");

  }
}
```

七、 局部变量

7.1 局部变量

> 概念：声明在方法内部的变量，必须先赋值再使用。

> 作用范围：定义行开始到所在的代码块结束。

注意：多个变量，在重合的作用范围内，不可出现重名（命名冲突）。

```
public class TestLocalVariable{

  public static void main(String[] args){

    int a = 10; //局部变量的声明，必须先赋值再使用

    System.out.println(a);


    if(1 == 1){

      int b = 20; //局部变量超出作用范围之后，会立即回收

      System.out.println(b);
    }

    //System.out.println(b);


    if(1 == 1){
      short a = 30;

      System.out.println(a);
    }


  }
}
```