# 第三章 选择

## 3.1引言

要点提示：程序可以基于条件决定执行哪些语句。

## 3.2 boolean数据类型

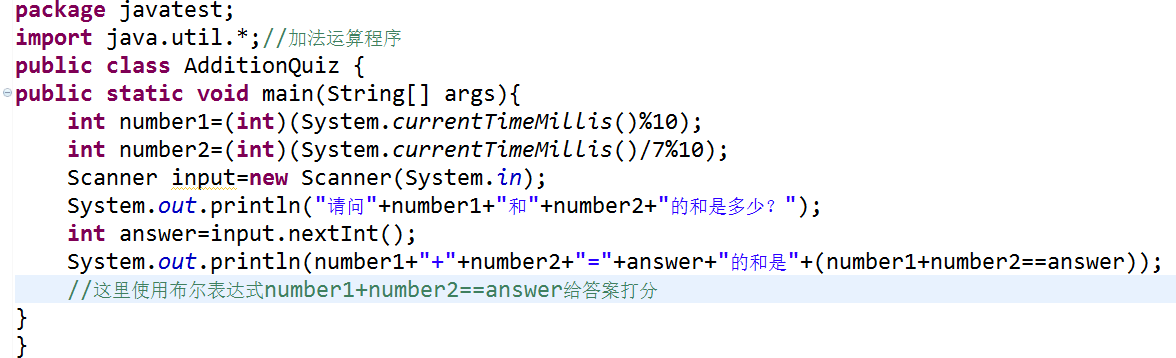
1.要点提示：boolean数据类型声明一个具有true或者false的变量。

2.警告：相等的关系操作符是两个等号（==），而不是一个等号（=），后者是指赋值操作符。

3.具有布尔值的变量称为“布尔变量”，boolean数据类型用于声明布尔型变量。Boolean型变量可以是以下这两个值中的一个：true和false。例如，下述语句将true赋值给变量lightsOn。

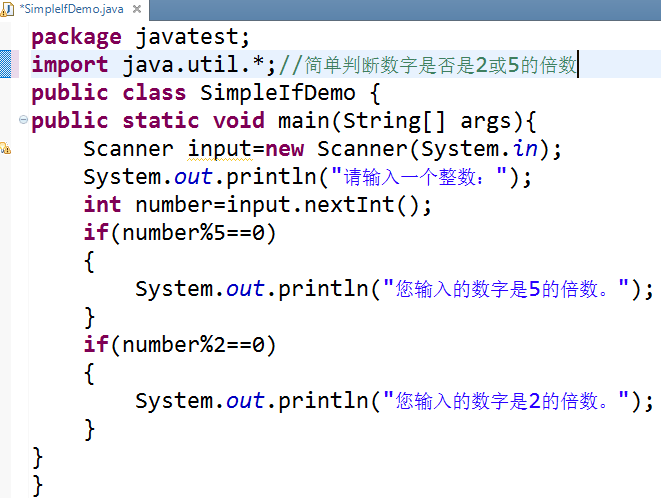
Boolean lightsOn=true;

True和false都是直接量，就像10这样的数字。它们被当作保留字一样，不能用于做程序中的标识符。



## 3.3 if语句

1.要点提示：if语句是一个结构，允许程序确定执行的路径。



## 3.4 双分支if-else语句

1.要点提示：if-else语句根据条件是真是假，决定执行的路径。

2.下面是双分支if-else语句的用法：

**if(布尔表达式)**

**{**

**布尔表达式为真时执行的语句（组）；**

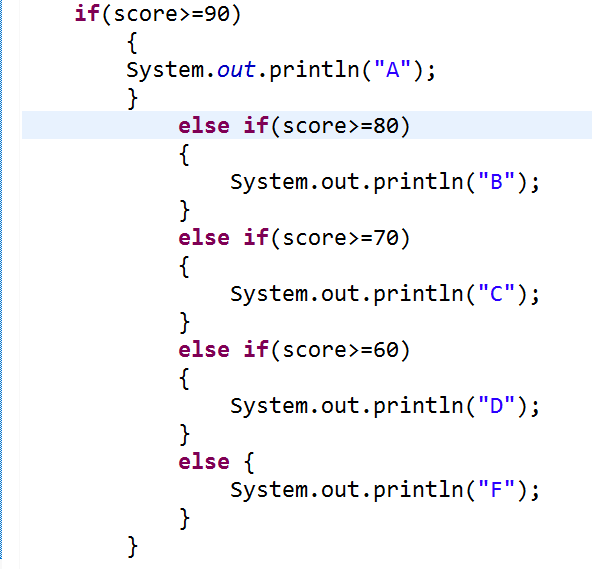
**}**

**else**

**{布尔表达式为假时执行的语句（组）；}**

## 3.5嵌套的if 语句和多分支if-else语句

1.要点提示：if语句可以在另外一个if语句中，形成嵌套的if语句。

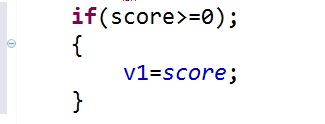


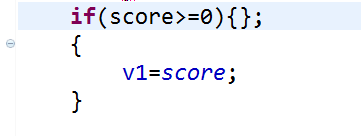
如图排版比较好，脉络更清晰。

## 3.6常见的错误和陷阱

**常见错误一：忘记必要的括号**

**常见错误二：在IF行出现错误的分号**

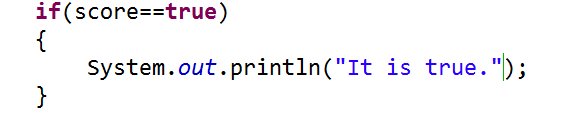




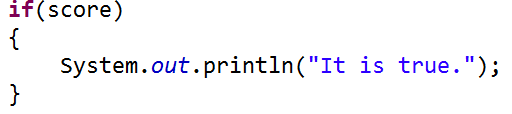
第一个图中的逻辑错误是很难发现的，它等价于第二个图中的形式。因为它既不是编译错误也不是运行错误，而是一个逻辑错误。

**常见错误三：对布尔值的冗余测试**

为了检测测试条件中的布尔类型变量是true还是false，像下图（a）中的使用相等比较操作符是多余的：



（a）



(b)

比较好的替代方法就是直接测试布尔变量，如图b这样，这么做的另一个原因就是避免出现难以发现的错误。

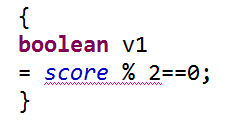
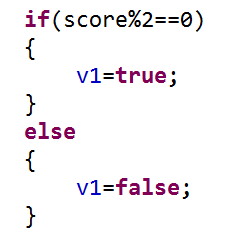
**常见错误四：悬空else出现的歧义**

Else和相对应的if需要对齐。在同一个块中，else会和最近的if匹配，为了强制else和距离远的if匹配，需要添加一对花括号。

**常见错误五：两个浮点数值的想等测试**

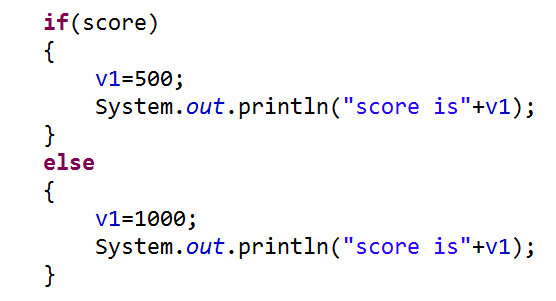
浮点数具有有限的计算精度，涉及涉及浮点数的计算可能引入取整错误。

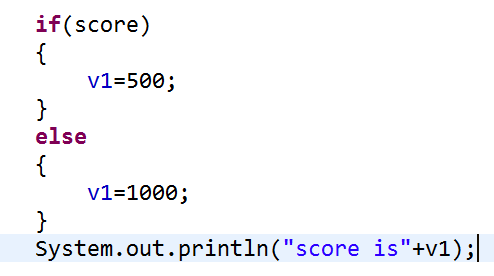
**常见陷阱一：简化布尔变量赋值**



两者等价，第二种形式要好。因为计算机执行效率更高吗

**常见陷阱二：避免不同情形中的重复代码**

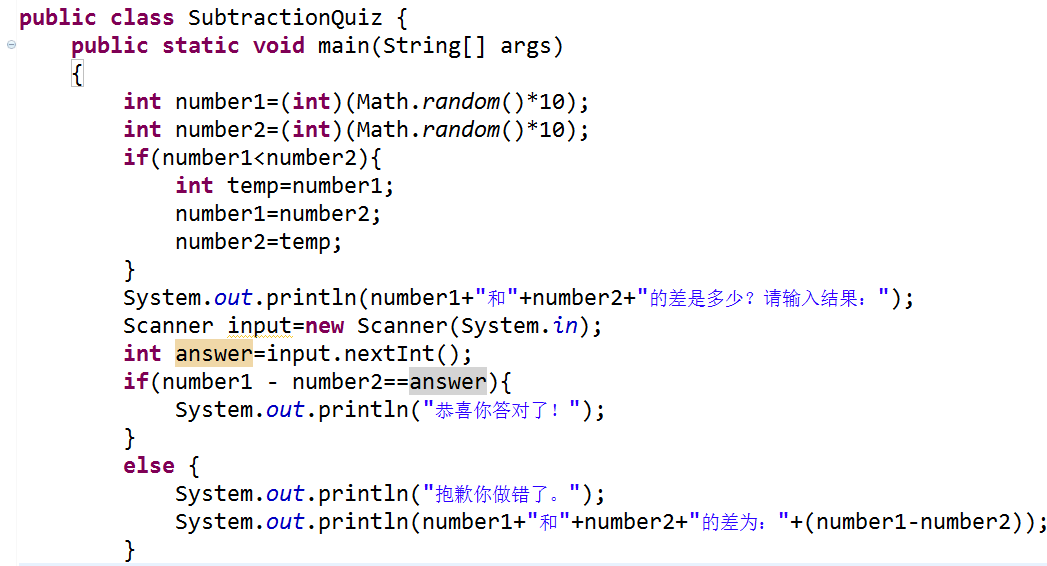




两种方法都是对的。新的代码去掉了重复代码，使得代码更容易维护，因为如果打印语句需要修改，你只需要修改一处地方。

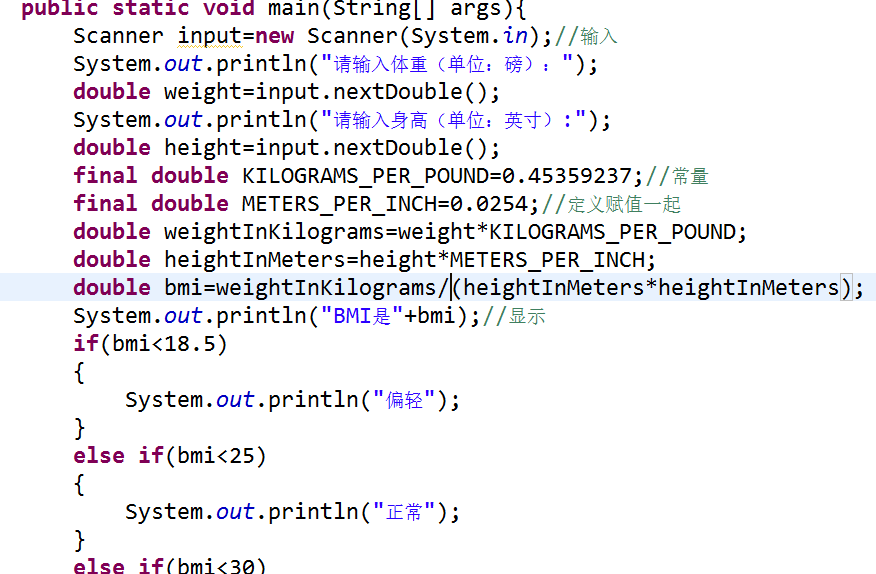
## 3.7产生随机数

1.要点提示：可以使用Math.random()来获取一个0.0到1.0之间的随机double值，不包括1.0。



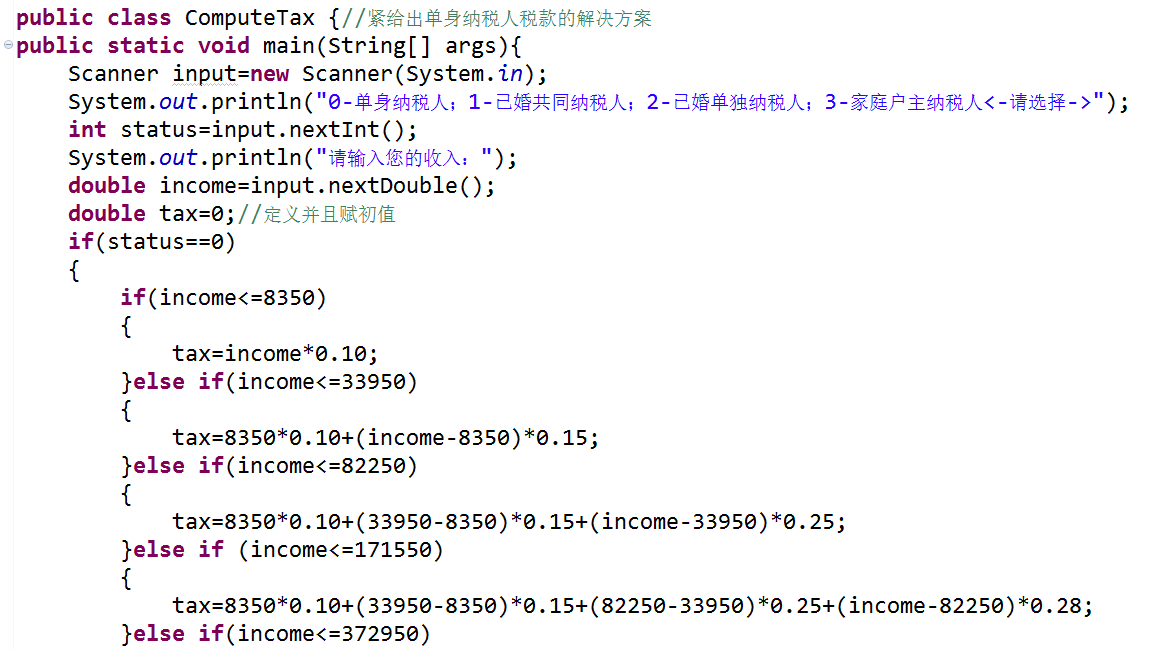
## 3.8 实例学习：计算身体质量指数

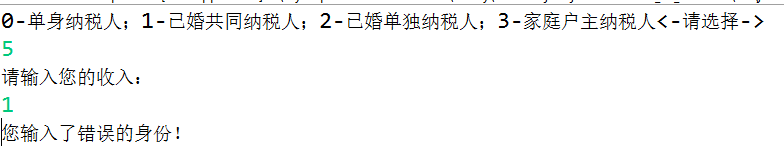
1.要点提示：使用嵌套If语句，计算身体质量指数。

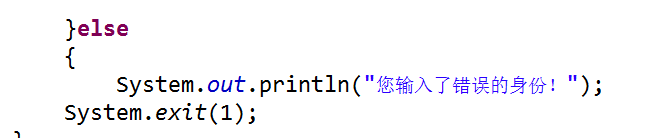


计算值不对

## 3.9示例学习：计算税率







1.一开始身份输入错误并没有报错，而是等到输入收入才显示身份错误。

2.System。Exit（1）；是啥意思？

3.提示：对所有的程序都应该先编写少量代码进行测试，之后再继续添加更多代码。这个过程称为“递进式开发与测试”。这种方法使得调试更加容易，因为错误很可能就在你刚刚添加的新代码中。

## 3.10 逻辑操作符

1.有时候，是否执行一条语句是由几个条件组合来决定的。可以使用逻辑操作符组合这些条件。

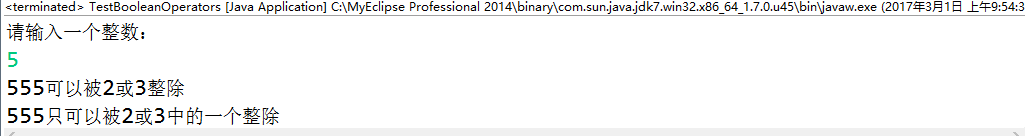
|  |  |  |
| --- | --- | --- |
| 操作符 | 名称 | 说明 |
| ！ | 非 | 逻辑非 |
| && | 与 | 逻辑与 |
| || | 或 | 逻辑或 |
| ˆ | 异或 | 逻辑异或 |

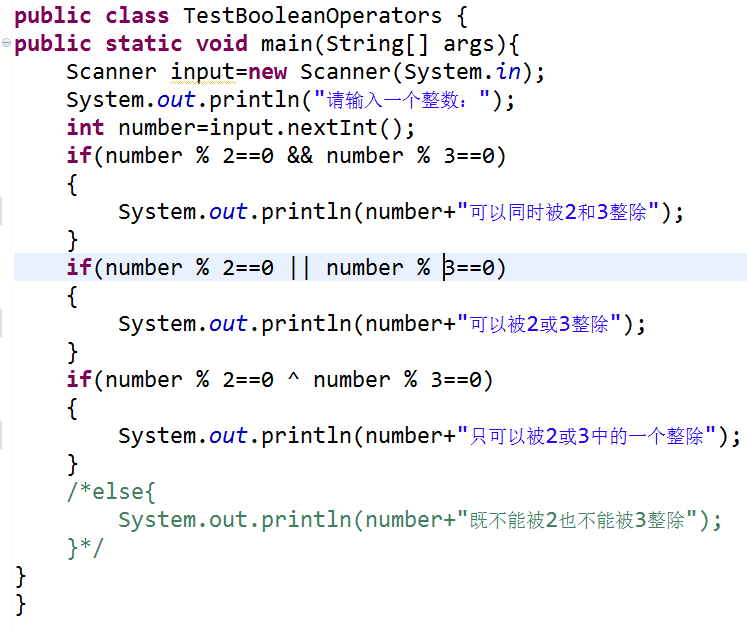
布尔操作符

2.异或表达式真值表

|  |  |  |
| --- | --- | --- |
| P1 | P2 | P1ˆP2 |
| false | false | False |
| false | true | True |
| true | false | True |
| true | true | false |

相同的false，不同的true。





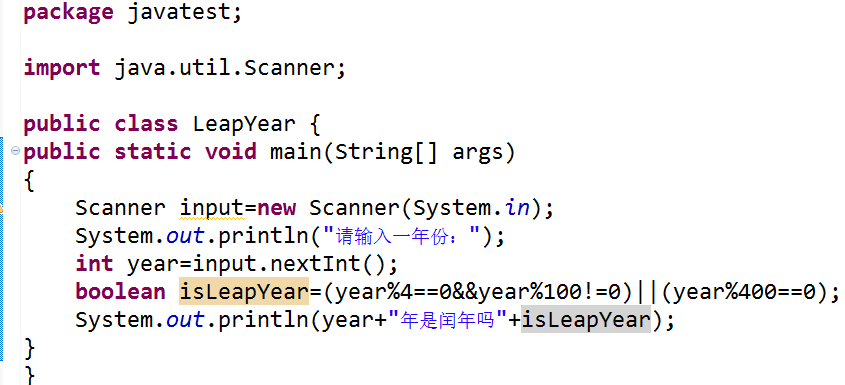
这个程序有三个问题：

其一，算不对；其二，555是什么鬼；其三，逻辑中为啥没有else

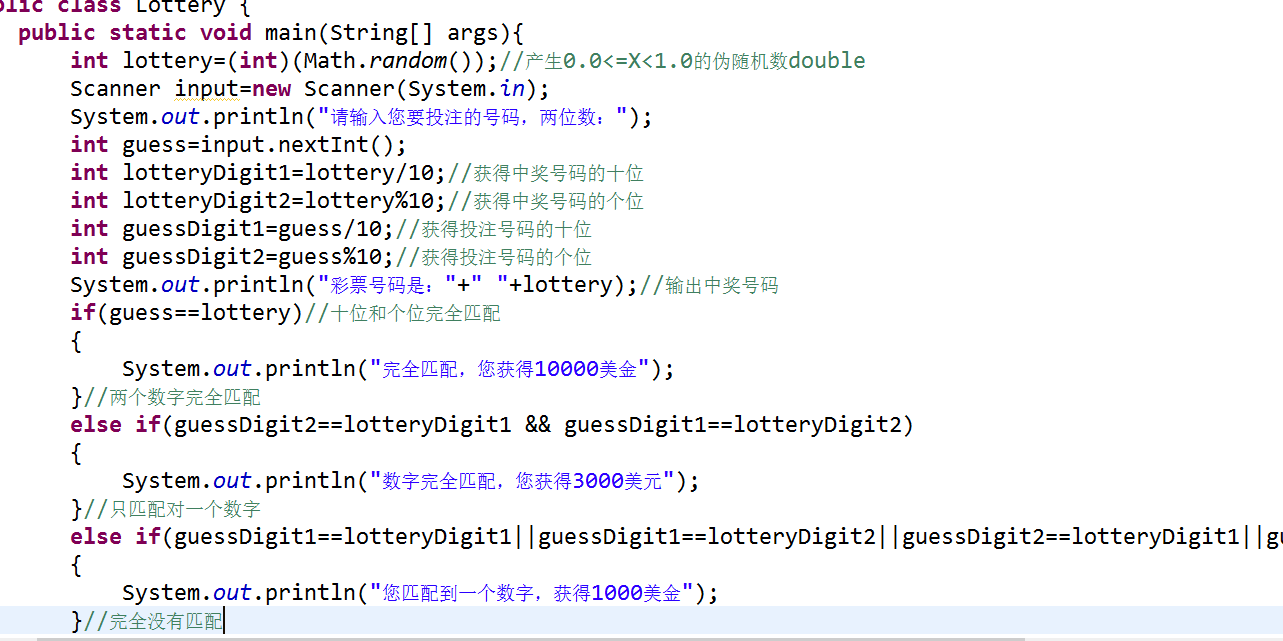
3.警告：从数学的角度看，表达式1<=number<=17是正确的，但是，在java中它是错误的，因为1<=num得到的是一个布尔值结果，它是不能和17进行比较的。这里的两个操作数（一个布尔值和一个数值）是不兼容的。正确的java表达式是：(1<=number)&&(number<=17)。

## 3.11示例学习：判定闰年

1.要点提示：如果某年可以被4整除而不能被100整除，或者可以被400整除，那么这一年便是闰年。



## 3.12 示例学习：彩票



注意：Math.random()产生“大于等于0.0，小于1.0的double值”

lottery/10获得十位；lottery%10获得个位。

## 3.13 switch语句

1. 要点提示：switch语句基于变量或者表达式的值来执行语句，Java提供的switch语句可以有效地来处理多重条件的问题。

2. 落空行为：一旦匹配其中一个case，就从匹配的case处开始执行，直到遇到break语句或一直到结束。例如：

switch（day）{

case 1:

case 2:

case 3:

case 4:

case 5: System.out.println(“工作日”);break;

case 0:

case 6:System.out.println(“周末”)；break；

上述代码，第一天到第五天显示“工作日”，周六周天显示“周末”。

3.根据年份判断生肖



## 3.14 条件表达式

1. 要点提示：条件表达式基于一个条件计算表达式的值。

例如：下面的语句在x大于0时给y赋值1；当x小于0时给y赋值-1。

If（x>0）

y=1;

else

y=-1;

将这个例子改成条件表达式，也能达到同样的效果。

y=（x>0）? 1 ：-1；

## 3.15操作符的优先级和结合规则

操作符优先级

## 3.16调试

1.要点提示：调试是在程序中找到和修改错误的过程。