**为了检测测试条件中的布尔型变量是true还是false，最好使用如下，避免冗余**

**if(even==true)换成if(even)**

**不能简单的用==判断两个浮点数值是否相等。可以通过测试两个数的差距小于某个阈（yu4）值，来比较它们是否已经足够接近。也就是，对于一个非常小的值**ε，如果|x-y|<ε,那么x和y非常接近。ε常用于表示一个非常小的值。通常将设为10^-14来比较两个double类型的值，而设10^-7来比较两个float类型的值。例如

final double EPSILON=1E-14;

double x=1.0-0.1-0.1-0.1-0.1-0.1;

if(Math.abs(x-0.5)<EPSILON)

System.out.println(x+"is approximately 0.5");

简化布尔变量赋值，例如

if(number%2==0)

even=true;

else

even=false;

替换成

boolean even=number%2==0；

**产生随机数**

你可以使用Math.random()来获得一个0.0到1.0之间的随机double值，不包括1.0.

(int)(Math.random()\*10)会返回一个随机的一位整数（即0-9之间的数）。

对所有的程序都应该先编写少量代码然后进行测试，之后再继续添加更多的代码。这个过程称为递进式开发和测试（incremental development and testing）。这种方法使得调试变得更加容易，因为错误很可能就在你刚刚添加进去的新代码中。

switch语句必须遵从下述规则：

1.switch表达式必须能计算出一个char、byte、short、int或者String型值，并且必须要用括号括住。

2.value1，....valueN必须与switch表达式的值具有相同的数据类型。value1，....valueN都是常量表达式，也就是说这里的表达式是不能包含变量的，例如，不能出现1+x。

3.当switch表达式的值与case语句的值相配时，执行从该case开始的语句，直到遇到一个break语句或到达该switch语句的结束。

4.默认情况（default）是可选的，当没有一个给出的case与switch表达式匹配时，用来执行该操作。

5.关键字break是可选的。break语句会立即终止switch语句。

如果故意省略break，在case子句后添加注释是一个好的做法。

**小结**

1.当对p1&&p2求值时，java先求p1的值，如果p1为true，再对p2求值；如果p1为false，就不再对p2求值。当对p1||p2求值时，java先求p1的值，如果p1为false，再对p2求值；如果p1为true，就不再对p2求值。因此，&&也称为条件与操作符或短路与操作符，而||也称为条件操作符或短路操作符。

2.switch语句根据char、byte、short、int或者String类型的switch表达式来进行控制决定。

3.括号可以强制求值的顺序以任何顺序进行。

4.除开赋值操作符的所有二元操作符都是左结合的，赋值操作符是右结合的。