7.3.2024第七天 书：java核心技术卷一

今日学习任务：学习自增自减运算符 还有关系和boolean运算符

3.5.6自增自减

在Java中，借鉴了C和C++中的做法，也提供了自增、自减运算符:

n++将变量n的当前值加1, n--则将n的值减1.例如，以下代码:

int n=12;

n++; n的值会变为13.

由于这些运算符会改变变量的值，所以不能对数值本身应用这些运算符。

例如，4++就不是一个合法的的语句。

这些运算符有两种形式:上面介绍的是运算符放在操作数后面的“后缀"形式。

还有一种前缀形式: ++n。后缀和前缀都会使变量值加1或减1.

但用在表达式中时，前缀会先完成加1;而后缀形式会使用变量原来的值。

比如看下面这段代码：

public class ZiZeng {  
 public static void main(String[] args){  
 int a = 1;  
 ++a;  
 System.*out*.println(a);  
 int b = 1;  
 b++;  
 System.*out*.println(b);  
  
 int c = 1;  
 int d = c++;/\*这个式子就等于  
 int d = c;  
 c++; 首先是d=c等于1 然后再去c++\*/  
 int e = ++c;//这个是直接等于c++后再去++c 等于3  
 System.*out*.println(d);//输出1  
 System.*out*.println(e);//输出3  
  
 int q = 1;  
 int w = ++q;  
 System.*out*.println(w);//输出2

int r = 1;  
int t = r++ \* 2;//这个式子等于  
//int t = r \* 2;相当于先把++隐藏了了  
//r++;  
System.*out*.println(t);// 输出2  
  
int y = 1;  
int s = ++y \* 2;  
System.*out*.println(s);//输出为4

}  
}

在Java中，很少在表达式中使用++。

3.5.7关系和boolean运算符

8个基本 byte short int long

Float double

Char 字符

Boolean 真假ture false

>=等于>或者== !=表示不相等 >和<=是相反关系

||是或者关系（n个关系中 只要从左到右 有一个是真 则全是真）

&&表示且的关系（且就是n个都要满足成立 一个是假 则就是假）

public class F1 {  
 public static void main(String [] args){  
 int a = 3;  
 int b = 4;  
 int c = 5;  
 System.*out*.println(a == b);//false  
 System.*out*.println(a != b);//ture  
 System.*out*.println(b == c);//false  
 System.*out*.println(b != c);//ture

boolean x = a > b;  
System.*out*.println(x);//false

}  
}

比如：

boolean x = a > b;  
 boolean x1 = b >= a;  
 boolean x2 = x || x1;  
 boolean x3 = x && x1;  
 boolean x4 = x && x1 && x3;  
 boolean x5 = x || x1 || x3;  
 boolean x6 = x && x1 || x3;  
 boolean x7 = x || x2 && x3;  
 System.*out*.println(x);  
 System.*out*.println(x1);  
 System.*out*.println(x2);  
 System.*out*.println(x3);  
 System.*out*.println(x4);  
 System.*out*.println(x5);  
 System.*out*.println(x6);  
 System.*out*.println(x7);  
 }  
}

||是或者关系（n个关系中 只要从左到右 有一个是真 则全是真）

&&表示并且的关系（且就是n个都要满足成立 一个是假 则就是假）

比如：

int a = 1;  
int b = 2;  
int c = 0;  
  
boolean b1 = b != 0 && a/b > 0;  
boolean b2 = c != 0 && b/c > 0;  
if (b1 && b2){  
 System.*out*.println("b1 && b2");  
}else{  
 System.*out*.println("not b1 && b2");  
}

3.5.5三木运算符

3.5.6switch表达式