# Java中>>和>>>移位操作符的区别

大家都知道>是比较两个对象的大小，那>>和>>>的区别呢？
>>和>>>都是移位操作；对正数的移位操作它们的功能都是一样的，如下：

15 >> 2 = 3
15 >>> 2 = 3

其实就是将15除于4，得到的商。转换为二进制可能更直观（为了方便，下面的二进制操作我们都是以八位进行的，而不是32位）：

0000 1111 >> 2 = 00000011 = 3，低位被丢失了
0000 1111 >>> 2 = 00000011 = 3，低位也被丢失了

对正数的操作它们的效果都是一样的，那么对于负数的移位是否也是一样呢？看下面例子就知道了：

-15 >> 2 = -4
-15 >>> 2 = 60

怎么会是这样的？负数的移位操作怎么变成正数了？同样我们将上面的式子转换为二进制来看看。首先我们得知道，在计算机中，负数是以补码的形式存储的（补码不知道？那你自己去好好学习点基础知识吧！）-15的补码是11110001，所以上面的操作转换为二进制如下所示：

11110001 >> 2 = 11111100（还是一个负数，转换为十进制就是-4）
11110001 >>> 2 = 00111100（这变成了正数了，转换成十进制就是60）

根据上面的结果，我们可以清楚的看出：

1、当移位的数是正数的时候，>> 和>>>效果都是一样的；
2、当移位的数是负数的时候，>>将二进制高位用1补上，而>>>将二进制高位用0补上，这就

导致了>>>将负数的移位操作结果变成了正数（因为高位用0补上了）。