- & (按位与)

&& (逻辑与)

| (按位或)

|| (逻辑或)

^ (异或运算符)

<< (左移运算符)

>> (右移运算符)

~ (取反运算符)

>>> (无符号右移运算符)

## 有符号右移>>

有符号右移就是右移之后，左边的补上符号位，正数补0，负数补1

## 无符号右移>>>

无符号右移就是右移之后，无论该数为正还是为负，右移之后左边都是补上0

## 左移<<

左移不区分有符号和无符号，都是左移之后右边补上0，最左边的符号位也直接移走

~取反运算

例子：

正数取反运算

```
15：0000 0000 0000 0000 0000 0000 0000 1111
取反运算（得到的是补码）：1111 1111 1111 1111 1111 1111 1111 0000
转原码：
1111 1111 1111 1111 1111 1111 1111 0000
                                   -1
1111 1111 1111 1111 1111 1111 1110 1111
取反得原码（符号位不变）：1000 0000 0000 0000 0000 0000 0001 0000
得 -16
```

负数取反运算（负数在计算机中以补码形式存储）

```
-15：1000 0000 0000 0000 0000 0000 0000 1111
转原码：
1000 0000 0000 0000 0000 0000 0000 1111
                                   -1
1000 0000 0000 0000 0000 0000 0000 1110
取反得原码（符号位不变）：1111 1111 1111 1111 1111 1111 1111 0001
取反运算：0000 0000 0000 0000 0000 0000 0000 1110
得 14
```

```JAVA
import java.util.*;

public class Main {
    public static void main(String[] args) {
        int i = 3&5; //按位与
        /** 3  011
         *  5  101
         *  i  001  1
         */
        int j = 3|5;//按位或
        /**  3 011
```

```
         *   5 101
         *   j 111   7
         */
        System.out.println(i);
        System.out.println(j);

        System.out.println((5 == 6) && (5 == 10));//逻辑与
        System.out.println((5 == 5) || (1 == 6));//按位或


        int h = 3^5;//按位异或
        /** 3  011
         *  5  101
         *  h  110  6
         *
         */
        System.out.println(h);

        System.out.println(h^h); //自己异或自己为0
        System.out.println(h^0);//任何数异或0 都为本身


        //技巧: 一个数组，里面的数只有一个出现一次，其它的出现2次，如何快速的找到这个数;
        int[] arr = {1,1,2,6,6,8,8,9,9};
        int a = 0;
        for(int m = 0; m < arr.length; m++){ //用零去异或数组的每一个元素，出循环后，单出现数即为a
            a ^= arr[m];
        }
        System.out.println(a);

        //右移:每右移一位除2
        System.out.println(10>>1);//5
        System.out.println(-2>>1);//-1
        //左移: 每左移一位乘2
        System.out.println(10<<1);//20
        System.out.println(-2<<1);//-4
        //箭头反向即为移位方向

        System.out.println(3>>>1);//1

        //按位求反(正数): 先写原码，在全部位求反，再求一次补码
        //负数:           先写原码，再求补码，最后全部位求反
        //练习1
        System.out.println(~10);
        // 原码 01010
        // 求反 10101
        //求补码 11011  -11
        //练习2
        System.out.println(~(-7));
        //原码   1111
        //求补码  1001
        //求反    0110
    }
```

```
}
```