链滴

详细解析位运算符 {~, ^, &, |} 和移位运算符{<<, >> , >>>}

作者：TWanGT

原文链接：https://ld246.com/article/1571194796459

来源网站：链滴

位运算符(~, ^, &, |)

~运算符(非)

位运算符, 每一位二进制都只和对应位的二进制进行计算

################ ~运算符(非) ################
十进制数字:2, 二进制:10
结果:十进制数字:-3, 二进制:11111111111111111111111111111101

解析: 将1改为0, 0改为1, 二进制:10前面隐藏了30个0

|运算符(或)

################ |运算符(或) ################
十进制数字:2, 二进制:10
十进制数字:8, 二进制:1000
结果:十进制数字:10, 二进制:1010

解析: 只要有1的位数结果就为1

^运算符(异或)

################ ^运算符(异或) ################
十进制数字:2, 二进制:10
十进制数字:8, 二进制:1000
结果:十进制数字:10, 二进制:1010

解析: 相同的位数结果为1, 不相同为0

相同: 即同为1或者同为0

&运算符(与)

################ &运算符(与) ################
十进制数字:2, 二进制:10
十进制数字:8, 二进制:1000
结果:十进制数字:0, 二进制:0

解析: 只有都为1的时候才为1, 其他情况都为0

移位运算符(<<, >> , >>>)

先普及两个概念,

Integer最大数:十进制数字:2147483647, 二进制:11111111111111111111111111111111

Integer最小数:十进制数字:-2147483648, 二进制:10000000000000000000000000000000

1. 99 移动 1位

<<运算符(左移)

################ <<运算符(左移) ################
效果: 二进制后面添加1个0
初始数:十进制数字:99, 二进制:1100011
移动位数:十进制数字:1, 二进制:1
十进制数字:198, 二进制:11000110

解析: 当移动位数为1时, 可近似看做是原数乘以2的1次方(类似10进制你后面每加一个0就扩大10倍), 是如果位数溢出(超过Integer最大数), 则结果就大不一样了(下面有例子)

>>运算符(右移)

################ >>运算符(右移) ################
效果: 二进制右边去掉5位, 左边补齐32位(按符号位补齐0或1)
初始数:十进制数字:99, 二进制:1100011
移动位数:十进制数字:1, 二进制:1
十进制数字:49, 二进制:110001

解析: 当移动位数为1时, 可近似看做是原数除2的1次方(丢弃余数), 如果除数比被除数还大, 就为0了

>>>运算符(右移忽视符号位)

################ >>>运算符(右移忽视符号位) ################
效果: 二进制右边去掉5位, 并且去掉最前面的符号位(按符号位补齐0或1并且补齐后将符号位改为0)
初始数:十进制数字:-99, 二进制:11111111111111111111111110011101
移动位数:十进制数字:1, 二进制:1
十进制数字:2147483598, 二进制:11111111111111111111111001110

解析: 去掉右边第一位, 并且将第32位(左边数第一位)改为0

2. -888888888 移动5位

################ <<运算符(左移) ################
效果: 二进制后面添加5个0
初始数:十进制数字:-888888888, 二进制:11001011000001001010000111001000
移动位数:十进制数字:5, 二进制:101
十进制数字:1620326656, 二进制:11000001001010000111001000000000

解析: 相当于乘以2的5次方, 由于整型位数为32位, 右边添加完5个0后为37位, 所以将最前面5位丢弃, 后10100000前面只剩下0, 所以最终结果是十进制:160

>>运算符(右移)

################ >>运算符(右移) ################
效果: 二进制右边去掉5位, 左边补齐32位(按符号位补齐0或1)
初始数:十进制数字:-888888888, 二进制:11001011000001001010000111001000
移动位数:十进制数字:5, 二进制:101
十进制数字:-27777778, 二进制:11111110010110000010010100001110

解析: 相当于除2的5次方, 右边去掉5位, 左边补齐5个1(如果本来第32为0则补全0)

>>>运算符(右移忽视符号位)

################ >>>运算符(右移忽视符号位) ################

效果: 二进制右边去掉5位, 并且去掉最前面的符号位(按符号位补齐0或1并且补齐后将符号位改为0)

初始数:十进制数字:-888888888, 二进制:11001011000001001010000111001000

移动位数:十进制数字:5, 二进制:101

十进制数字:106439950, 二进制:1100101100000100101000001110

解析: 右边去掉5位, 左边补齐5个0

示例源码

```
package com.wang.math;

/**
 * @Author: WanG
 * @Date: 2019-02-12 15:39
 * @version: v1.0
 * @description: 位运算符
 */
public class 位运算符 {


    public static void main(String[] args) {
        printInfo("整型最大数:", Integer.MAX_VALUE);
        printInfo("整型最小数:", Integer.MIN_VALUE);
        位运算();
        移位运算();
    }
    public static void 位运算() {
        int before = 2;
        int before2 = 8;

        System.out.println("\n################  ~运算符(非) ################");
        printInfo(before);
        printInfo("结果:", ~before);

        System.out.println("\n################  |运算符(或) ################");
        printInfo(before);
        printInfo(before2);
        printInfo("结果:", before|before2);

        System.out.println("\n################  ^运算符(异或) ################");
        printInfo(before);
        printInfo(before2);
        printInfo("结果:", before^before2);

        System.out.println("\n################  &运算符(与) ################");
        printInfo(before);
        printInfo(before2);
        printInfo("结果:", before&before2);
```

```
    }

    /**
     * 将符号位直接移动
     * 当移动位数为1时, 接近乘以二和除2的效果
     * @author wangq 2019-02-12 15:56
     * @param
     * @return
     */
    public static void 移位运算() {
        int before = -99;
        int before2 = 5;

        System.out.println("\n#################  <<运算符(左移) #################");
        System.out.println("效果: 二进制后面添加" + before2 + "个0");
        printInfo("初始数:", before);
        printInfo("移动位数:", before2);
        printInfo(before<<before2);


        System.out.println("\n#################  >>运算符(右移) #################");
        System.out.println("效果: 二进制右边去掉" + before2 + "位, 左边补齐32位(按符号位补齐0或1
");
        printInfo("初始数:", before);
        printInfo("移动位数:", before2);
        printInfo(before>>before2);

        System.out.println("\n#################  >>>运算符(右移忽视符号位) ############
###");
        System.out.println("效果: 二进制右边去掉" + before2 + "位, 并且去掉最前面的符号位(按符
位补齐0或1并且补齐后将符号位改为0)");
        printInfo("初始数:", before);
        printInfo("移动位数:", before2);
        printInfo(before>>>before2);

    }

    /**
     * 打印
     * @author wangq 2019-02-12 15:40
     */
    private static void printInfo(int num) {
        printInfo("", num);
    }

    /**
     * 打印
     * @author wangq 2019-02-12 15:40
     */
    private static void printInfo(String msg, int num) {
        System.out.println(String.format(msg + "十进制数字:%s, 二进制:%s", num, Integer.toBinar
String(num)));
    }
}
```