Number Complement

作者：moloee
原文链接：https://ld246.com/article/1502946238880
来源网站：链滴

Given a positive integer, output its complement number. The complement strategy is to flip t e bits of its binary representation.

给定一个 unsigned int 类型，返回按位取反的结果。

基本思路：直接按位取反？发现不行，会发现，num=5, ~num= -6，为啥呢。。

这里涉及到一个问题，java 中没有 unsigned 这个类型，int 类型的存储是以二进制的补码存储的，数的补码和反码是其本身，但是负数不是。

- num=5, => 101，存储的时候最高位加一位符号位，即 0101，[0正1负]
- 按位取反后得到 1010，这是负数的补码，转换为10进制，需要 -1 后，除符号位外 按位取反，得到 1001 => 1110[^complement]. 也就是 -6 了。

所以问题来了，原本比较简单的问题，现在因为符号位的问题不太好处理了。当然如果生硬的按位判然后再拼一个结果，倒也不是不可以，毕竟也没意思。

在 discuss 里看到了一个方案。其实我们最需要解决的就是按位取反之后怎么把 1010 的最高位置 0 问题。

解决方案就是，创建一个跟 1010 位数一样多的二进制，而最高位是 0 其他位是 1，即 0111，然后跟 1010 按位与，这样就可以保持其他位不变，最高位置反。Integer 类中有个方法 highestOneBit，理还有一个 lowestOneBit，通过这个最高位可以通过如下方式达到目的：

```
(Integer.highestOneBit(num) << 1) - 1
(~num) & (Integer.highestOneBit(num) << 1) - 1 //这样即满足条件
```

看了下highestOneBit/lowestOneBit的实现，

```
public static int highestOneBit(int i) {
    // HD, Figure 3-1
    i |= (i >>  1);
    i |= (i >>  2);
    i |= (i >>  4);
    i |= (i >>  8);
    i |= (i >> 16);
    return i - (i >>> 1);
}

public static int lowestOneBit(int i) {
    // HD, Section 2-1
  return i & -i;
}
```

[^complement]：补码，负数->补码的计算，-5的源码是 1101, 除符号位外，按位取反 1010，得反码，然后+1得到补码 1011.

原文链接：Number Complement