数字的补数

给你一个 **正** 整数 num ，输出它的补数。补数是对该数的二进制表示取反。

示例 1：

输入：num = 5

输出：2

解释：5 的二进制表示为 101（没有前导零位），其补数为 010。所以你需要输出 2 。

示例 2：

输入：num = 1

输出：0

解释：1 的二进制表示为 1（没有前导零位），其补数为 0。所以你需要输出 0 。

5 的二进制是：0101，7 的二进制是： 0111，它们的抑或为：0010，去掉前导零位即为取反。
再来一个例子，假设 a 为 1110 0101，b 为 1111 1111，a^b = 0001 1010 是 a 的取反。也就是说二进制位数与 num 相同，且全为 1 的数 tmp 与 num 的抑或即为所求。

```
class Solution {
public:
    int findComplement(int num) {
        //与全为 1 的异或即可得到补码
        //全为 1 的数必须保证与 num 二进制位数长度相同
        int tmp=1;
        while(tmp<num)
        {
            tmp<<=1;//不够长左移 补位为 1
            tmp++;
        }
        return num^tmp;
    }
};
```