httpsmp.weixin.qq.comsufQdFDFR3oQsG\_WyMq35hQ

# 盘点：这 7 道位运算面试题，哪一题你能做对？

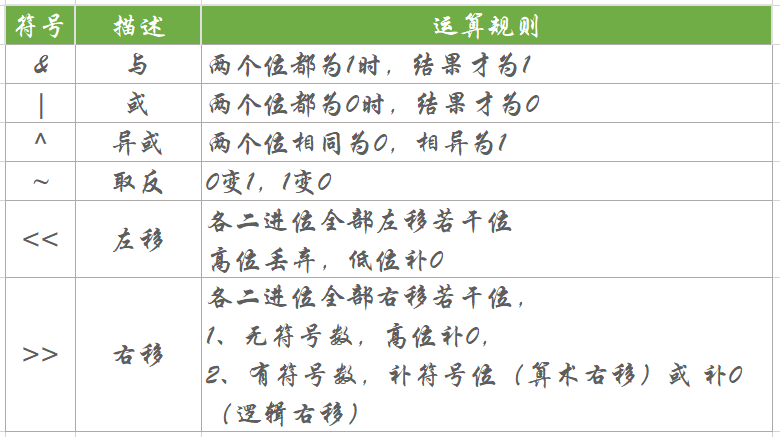
以下是目录，列出的全部内容都应该进行掌握：

* 位运算基础
* 位运算的奇淫技巧
* 两数之和
* 二的幂
* 一的个数
* 只出现一次的数字Ⅰ
* 只出现一次的数字Ⅱ

## 位运算基础

程序中的所有数在计算机内存中都是以二进制的形式储存的，位运算就是直接对整数在内存中的二进制位进行操作。

首先我们还是简单列下常规的位运算：



## 位运算的奇淫技巧

上面的内容相对比较常规，**但是一般面试我们遇到的，都不是常规内容**。所以下面这些，是必须掌握的。

下面的这八个技巧，基本cover了位运算90%的面试题：

1、使用 x & 1 == 1 判断奇偶数。（注意，一些编辑器底层会把用%判断奇偶数的代码，自动优化成位运算）

2、不使用第三个数，交换两个数。x = x ^ y ， y = x ^ y ， x = x ^ y。（早些年喜欢问到，现在如果谁再问，大家会觉得很low）

3、两个相同的数异或的结果是 0，一个数和 0 异或的结果是它本身。（对于找数这块，异或往往有一些别样的用处。）

4、x & (x - 1) ，可以将最右边的 1 设置为 0。（这个技巧可以用来检测 2的幂，或者检测一个整数二进制中 1 的个数，又或者别人问你一个数变成另一个数其中改变了多少个bit位，统统都是它）

5、异或可以被当做无进位加法使用，与操作可以用来获取进位。

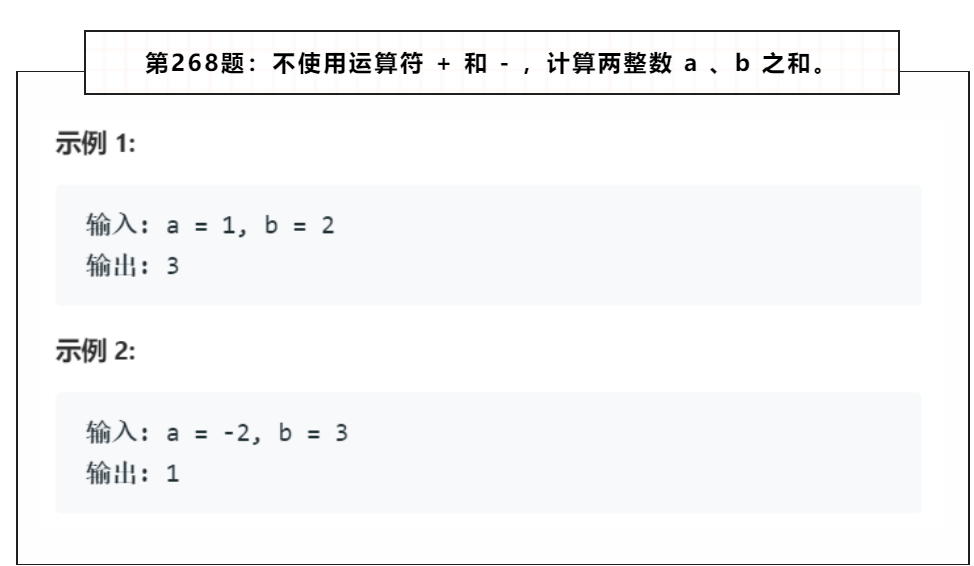
6、i+(~i)=-1，i 取反再与 i 相加，相当于把所有二进制位设为1，其十进制结果为-1。

7、对于int32而言，使用 n >> 31取得 n 的正负号。并且可以通过 (n ^ (n >> 31)) - (n >> 31) 来得到绝对值。（n为正，n >> 31 的所有位等于0。若n为负数，n >> 31 的所有位等于1，其值等于-1）

8、使用 (x ^ y) >= 0 来判断符号是否相同。（如果两个数都是正数,则二进制的第一位均为0,x^y=0；如果两个数都是负数,则二进制的第一位均为1；x^y=0 如果两个数符号相反,则二进制的第一位相反,x^y=1。有0的情况例外，^相同得0，不同得1）

## 两数之和

从最简单的开始讲起。这个题很老了，拿出来给不会的同学看一看，会的直接跳过。（值得一说的是，这个题目在国外上，有2000个dislike，可以看到大家的嫌弃！）

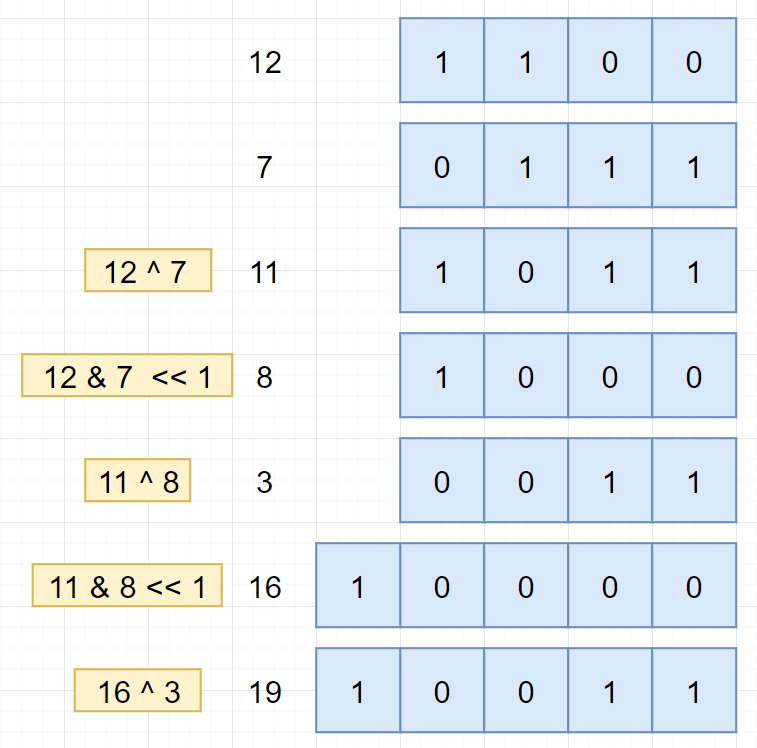


直接使用上面我们讲过的奇淫技巧进行解题：

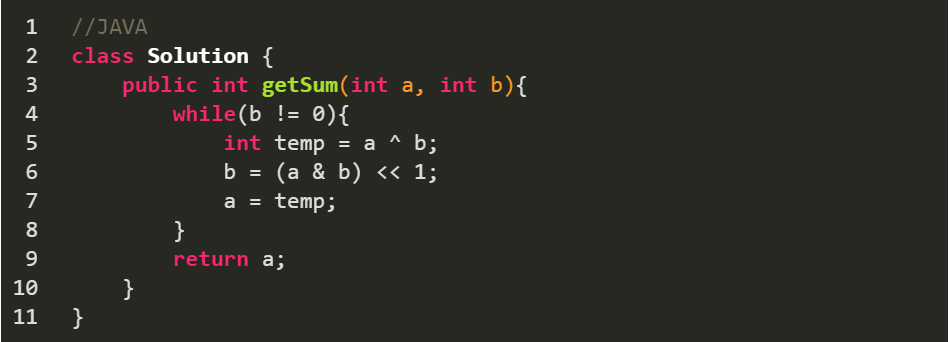
“异或”是一个**无进位加法**，说白了就是把进位砍掉。比如01^01=00。

“与”**可以用来获取进位，**比如01&01=01，然后再把结果左移一位，就可以获取进位结果。

根据上面两个技巧，假设有 12+7：



根据分析，完成题解：

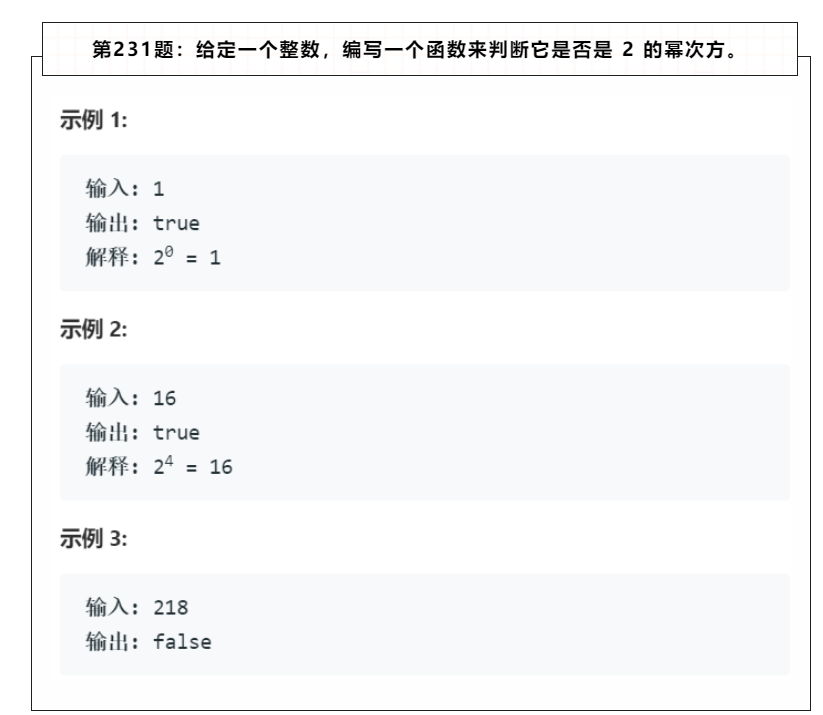


看不懂，参考下力扣的题解吧：

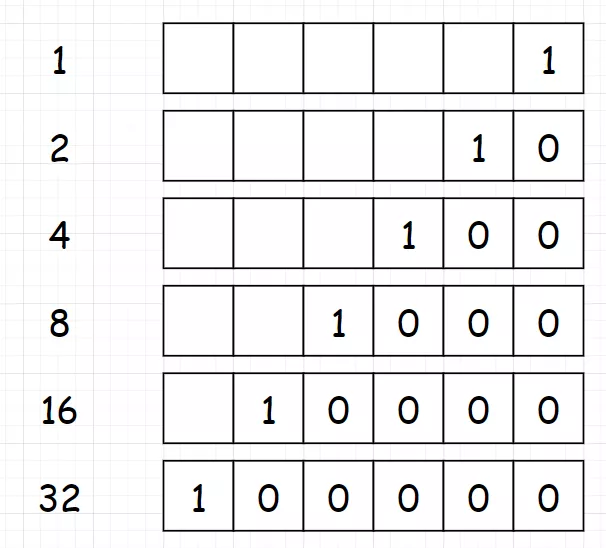
<https://leetcode-cn.com/problems/bu-yong-jia-jian-cheng-chu-zuo-jia-fa-lcof/solution/er-jin-zhi-qiu-he-chao-xiang-xi-da-bai-10000yong-h/>

## 2的幂

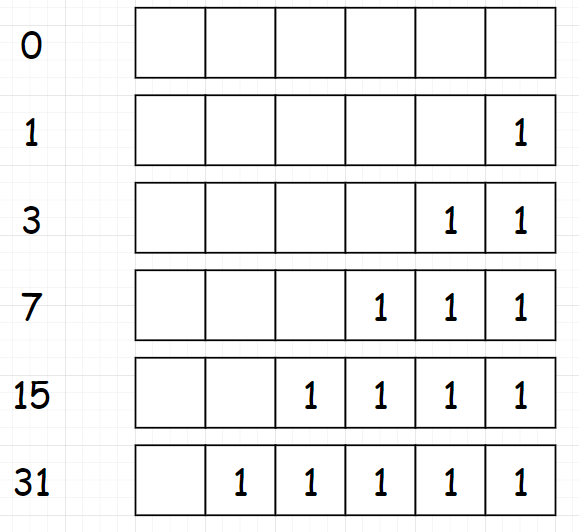
做这道题前，可以翻到最前面，看一看可以使用哪一个技巧。找到了，你就会了。



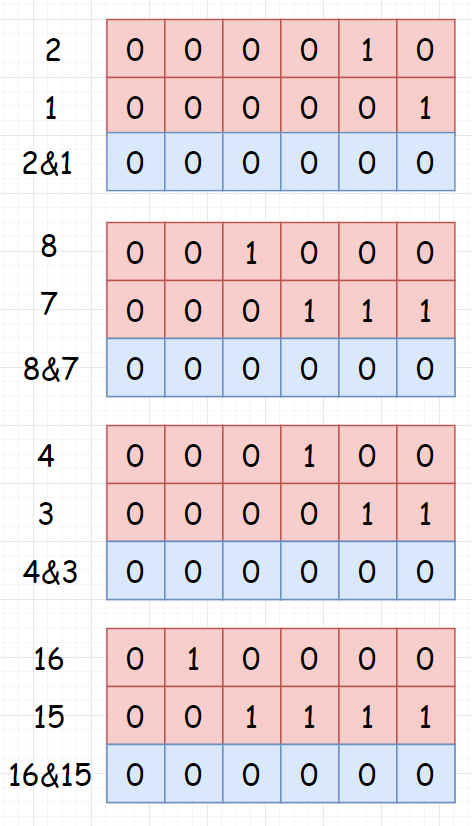
先观察一些是2的幂的二进制数：



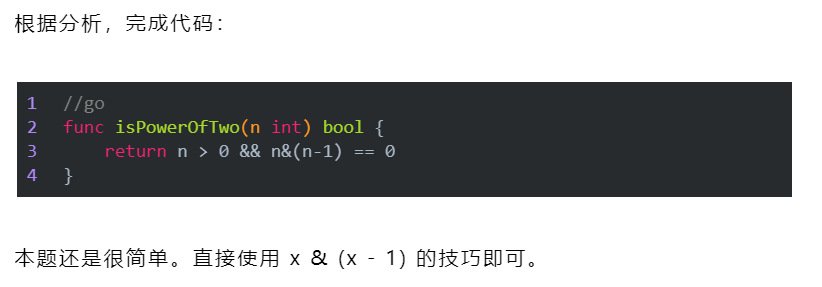
**可以发现这些数，都是最高位为1，其他位为0**。所以我们把问题转化为“判断一个数的二进制，除了最高位为1，是否还有别的1存在”。然后我们再观察下面这样的一组数，对应着上面的数减去1：



我们对两组数求“&”运算：

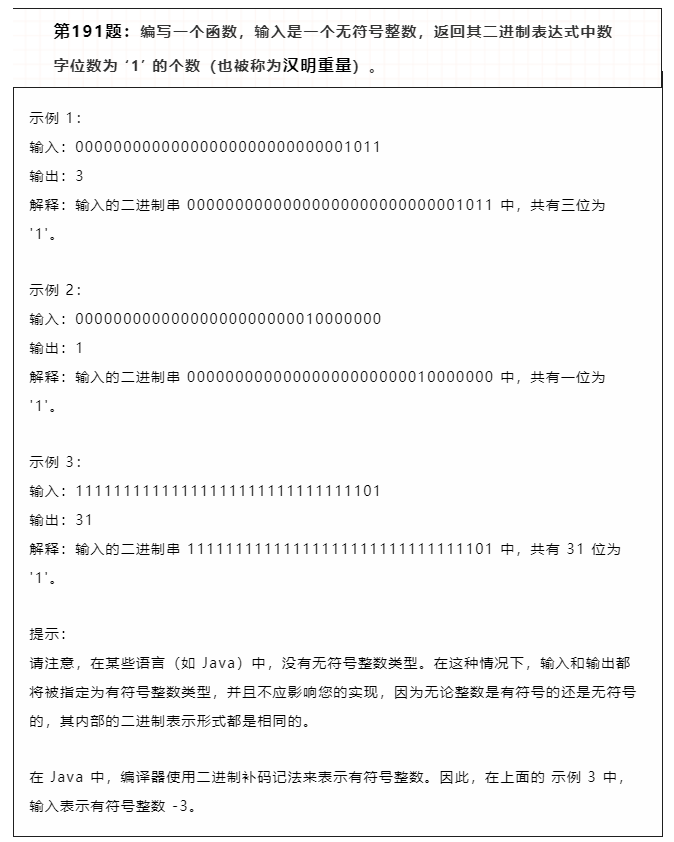


可以看到，对于N为2的幂的数，**都有 N&(N-1)=0 ，**所以这就是我们的判断条件。（这个技巧可以记忆下来，在一些别的位运算的题目中也是会用到的）



## 一的个数

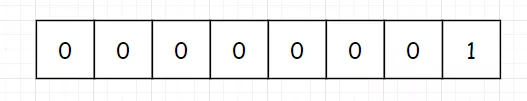
略微增大一点难度，讲这道题目意义是引入一个概念“掩码”。掩码是指使用一串二进制代码对目标字段进行位与运算，屏蔽当前的输入位。



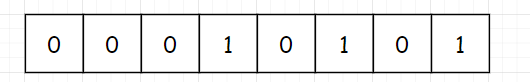
题目稍微长了点，但是我之前说过。对于大部分的题而言，题目越长，越简单。

首先最容易想到的方法是：**我们直接把目标数转化成二进制数，然后遍历每一位看看是不是1，如果是1就记录下来**。通过这种比较暴力的方式，来进行求解。比如Java中，int类型是32位，我们只要能计算出当前是第几位，就可以顺利进行求解。

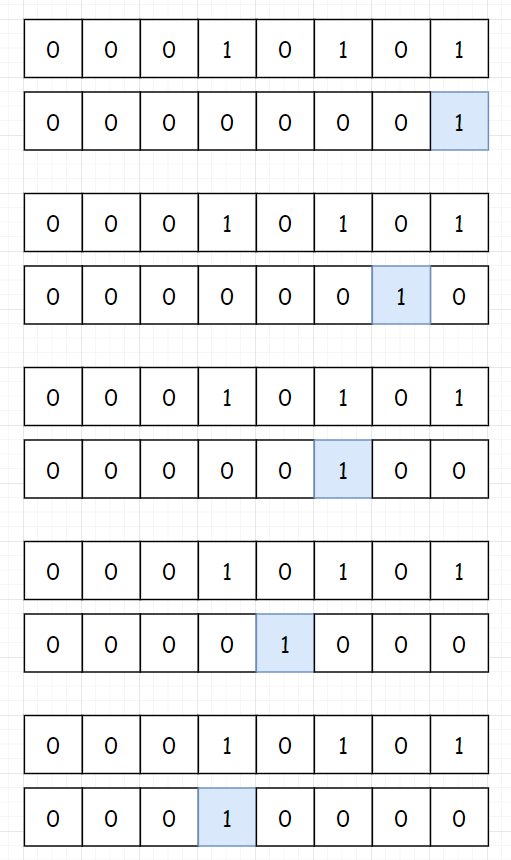
那如何计算当前是第几位呢，我们可以构造一个掩码来进行，说掩码可能大家听着有点懵逼，其实就是弄个1出来，1的二进制是这样：



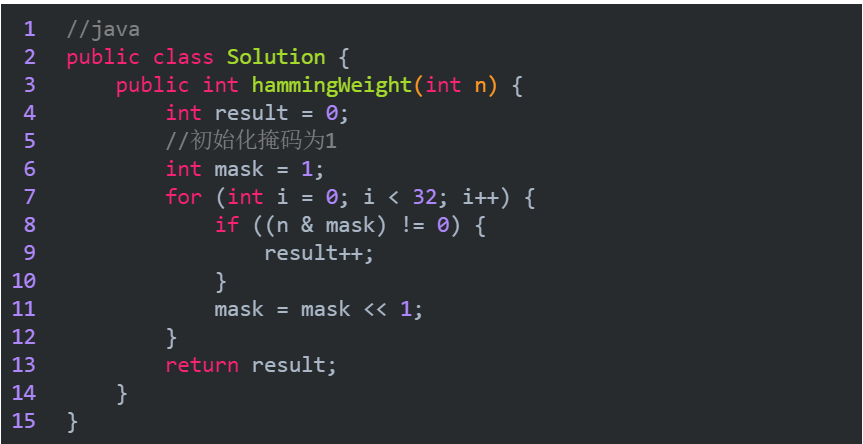
我们只需要让这个掩码每次向左移动一位，然后与目标值求“&”，就可以判断目标值的当前位是不是1。比如目标值为21，21的二进制是这样：



然后每次移动掩码，来和当前位进行计算：



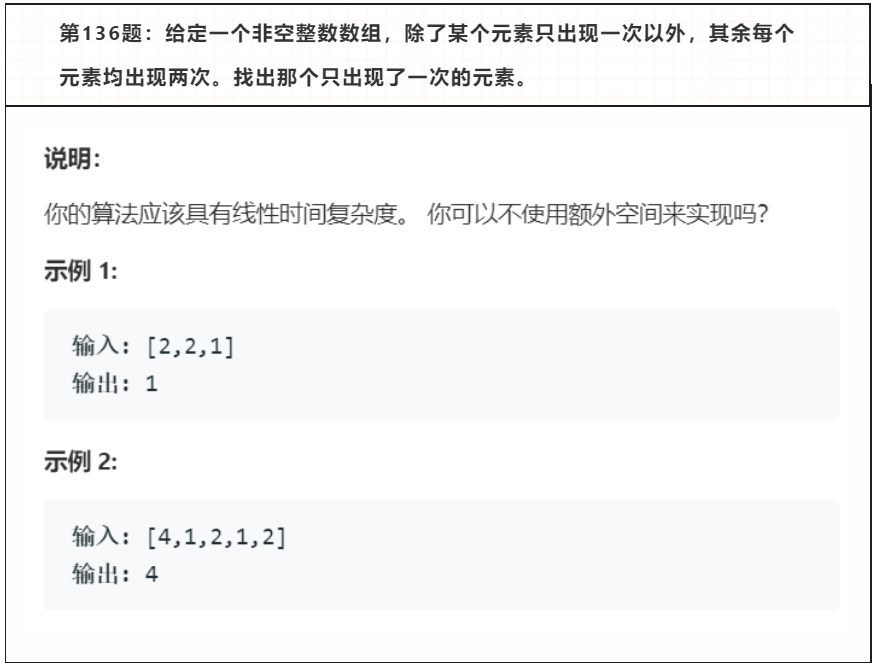
根据分析，完成代码：



唯一需要提醒的地方是：判断 n&mask 的时候，不要错写成 (n&mask) == 1，因为这里你对比的是十进制数。新人很容易犯这样的错误。

## 只出现一次的数字

我们再稍微提高一点难度。大家想想用什么思路进行求解？



直接分析，**我们要找只出现一次的数字，并且已知了其他的数字都只出现了两次。**那么这种一听其实就应该想到需使用**位运算**来进行求解。最好的，就是在读完题目的瞬间，直接条件反射！（当然，如果你现在第一反应是想到 通过遍历统计，或者其他如使用hashmap 等方式来进行求解，那我觉得你的位运算这块，是有必要加强练习力度的。如果你第一反应，连思路都没有，那我觉得对于整个算法的能力这块，都是比较欠缺的，需要下苦功！）

回到题目，如何使用位运算进行求解呢？对于任意两个数a和b，我们对其使用 “**异或**”操作，应该有以下性质：

* 任意一个数和0异或仍然为自己：

*a*⊕0=*a*

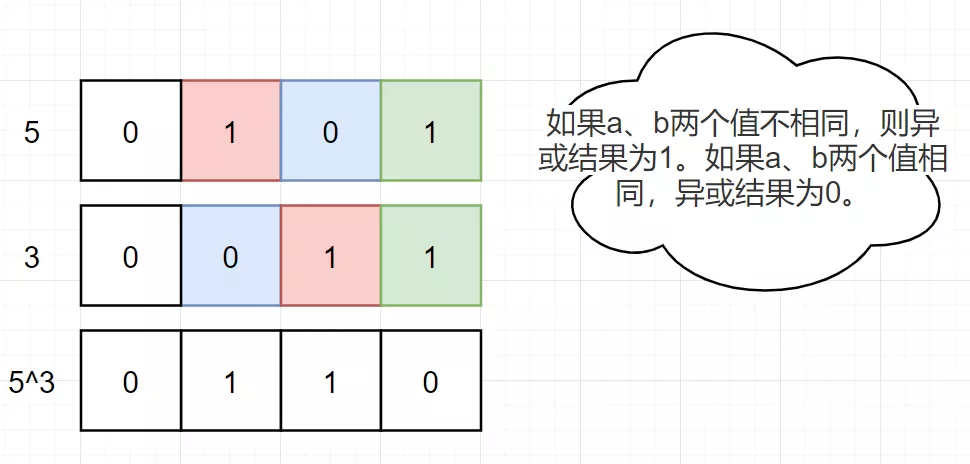
* 任意一个数和自己异或是0：

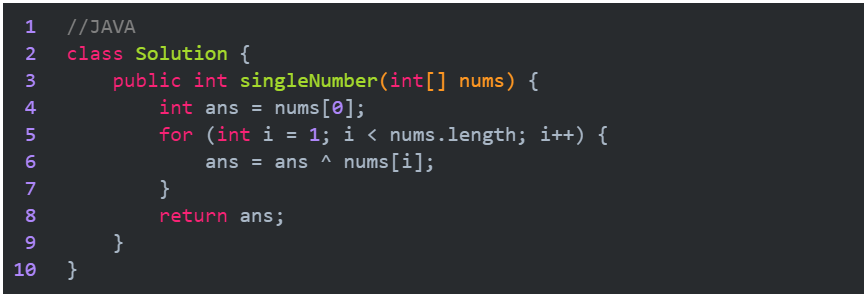
a⊕a=0

* 异或操作满足交换律和结合律：

a⊕b⊕a=(a⊕a)⊕b=0⊕b=*b*

可能有人直接都不知道异或是什么，所以还是举个例子，比如5异或3，也就是5⊕3，也就是5^3，是下面这样：





## 只出现一次的数字 Ⅱ

## 