剑指56-I 数组中数字出现的次数

性质：异或 具有交换性。a^b^c... 顺序可以随意排列。
2个相同的值 异或 后 为0。
ps：这个题目用hash也可以。

让我们先来考虑一个比较简单的问题：
如果除了一个数字以外，其他数字都出现了两次，那么如何找到出现一次的数字？

答案很简单：全员进行异或操作即可。考虑异或操作的性质：对于两个操作数的每一位，相同结果为 0，不同结果为 1。那么在计算过程中，成对出现的数字的所有位会两两抵消为 0，最终得到的结果就是那个出现了一次的数字。

那么这一方法如何扩展到找出两个出现一次的数字呢？
如果我们可以把所有数字分成两组，使得：
A.两个只出现一次的数字在不同的组中；
B.相同的数字会被分到相同的组中。
那么对两个组分别进行异或操作，即可得到答案的两个数字。这是解决这个问题的关键。
全员异或值的某一位是1，代表目标2个数的那一位是不同的。那么就根据这1位是0还是1来分为2组就能满足A。同时如果两个数相同，那不可能存在有1位是不同的，这样的分组保证了B。

```
// 别卷啦别卷啦！！！！
class Solution {
public:
    vector<int> singleNumbers(vector<int>& nums) {
        int tmp = nums[0];
        for(int i = 1; i < nums.size(); i++) {
            tmp = tmp^nums[i];
        }

        int flag = 1;
        while((tmp & flag) == 0) {
            // 从低位到高位寻找第一个为1的位，这就是原数组中两个目标值出现分歧的位，根据这个分歧后面可以把数组分成两个组
            flag <<= 1;
        }

        int first = 0, second = 0;
        for(int i:nums) {
            if(i & flag) {
                first ^= i;
            }
            else {
                second ^= i;
            }
```

```
        }
        return vector<int> {first, second};
    }
};
```