只出现一次的数字 III

给定一个整数数组 nums，其中恰好有两个元素只出现一次，其余所有元素均出现两次。 找出只出现一次的那两个元素。你可以按 **任意顺序** 返回答案。

进阶：你的算法应该具有线性时间复杂度。你能否仅使用常数空间复杂度来实现？

示例 1：

输入：nums = [1,2,1,3,2,5]

输出：[3,5]

解释：[5, 3] 也是有效的答案。

示例 2：

输入：nums = [-1,0]

输出：[-1,0]

示例 3：

输入：nums = [0,1]

输出：[1,0]

```
class Solution {
public:
    vector<int> singleNumber(vector<int>& nums) {
        vector<int> res;
        //先对所有数字进行一次异或，得到两个出现一次的数字的异或值
        int ret=0;
        for(auto& n:nums)
        {
            ret^=n;
        }
        //在异或结果中找到任意为 1 的位
        int div=1;
        while((div&ret)==0)
            div<<=1;
        /*
        根据这一位对所有的数字进行分组。
```

```
        在每个组内进行异或操作，得到两个数字
        */
        int n1=0;
        int n2=0;
        for(auto& n:nums)
        {
            if(div&n)
            {
                n1^=n;
            }
            else
            {
                n2^=n;
            }
        }
        res.push_back(n1);
        res.push_back(n2);
        return res;
    }
};
```