260. Single Number III

题目描述：https://leetcode.com/problems/single-number-iii/

> 给定一个数组，这个数组中只有两个数只出现一次，其他都出现两次，求这两个数。顺序无所谓。
> 例如：
>
> ```
> 0, 0, 1, 2
> ```
>
> 返回：
>
> ```
> 1, 2
> ```

解题思路：

首先我们由Single Number I得知，我们可以通过异或操作拿到出现奇数次的数字，但是结果只能是A xor B，现在的问题就是我们如何才能单独取出A或者B？
首先我们可以找到A和B不同的地方，A xor B结果是A和B不相同的位组成的数字。
我们可以拿出某一位不相同的bit，然后根据A & bit == 0 那么 B & bit == 1 或者反过来，来区分开A和B来做异或操作。

代码：

```
class Solution {
public:
    vector<int> singleNumber(vector<int>& nums) {
        int aXorB = 0;
        int resA = 0, resB = 0;
        for(int i = 0; i < nums.size(); i++){
            aXorB ^= nums[i];
        }
        int LastDiffBitAandB = (aXorB & (aXorB - 1)) ^ aXorB;
        for(int i = 0; i < nums.size(); i++){
            bool res = nums[i] & LastDiffBitAandB;
            if(!res){
                resA = resA ^ nums[i];
            }
            else{
                resB = resB ^ nums[i];
            }
        }
        return vector<int>{resA, resB};
    }
};
```