链滴

算法 01 异或运算

作者：AshShawn

原文链接：https://ld246.com/article/1646119266747

来源网站：链滴

一 基本概念

定义: 位运算中, 相同为0,不同为1,即1^0=1

实际可以记为 无进位相加.

二 性质

性质一:　　0^N == N

N^N == 0　(N为整数)

性质二:　满足交换律和结合律

A^B=B^A

(A^B)^C=A^(B^C)

三 演练

1. 交换两个数

```
private void swap(int[] arr, int i, int j) {
    arr[i] = arr[i] ^ arr[j]; //
    arr[j] = arr[i] ^ arr[j]; // arr[i] ^ arr[j] ^ arr[j]=arr[i] ^ 0=arr[i]
    arr[i] = arr[i] ^ arr[j]; //  arr[i] ^ arr[j] ^ arr[j]=0 ^ arr[j]=arr[j]
}
```

2. 奇数次与偶数次问题

问题描述: 一个数组中有一种数出现了奇数次，其他数都出现了偶数次，怎么找到并打印这种数

```
/**
 * 查找出奇数次的数
 */
public int findOddTimesNumber(int[] arr) {
    int ans = 0;
    for (int i = 0; i < arr.length; i++) {
        ans ^= arr[i];
    }
    return ans;
}
```

3. 提取最右位的1

```
    /**
     * 原值: 0 1 0 1 0 0 0 0
     * 取反: 1 0 1 0 1 1 1 1
     * +1 : 1 0 1 1 0 0 0 0
     * &= : 0 0 0 1 0 0 0 0
```

```
     */
    public int findRightBit1(int num) {
//        return num & (~num + 1);  //取反后+1
        return num & (-num);        //负值
    }
```

4. 奇数次与偶数次问题Plus

问题: 一个数组中有两种数出现了奇数次，其他数都出现了偶数次，怎么找到并打印这两种数

分析:

1. 假设这两个出现奇数次的数位a,b ;
2. 将所有数做异或操作,得到的值肯定为 a^b;
3. 找到最后一位1所在的位置index , 即a和b 在该位一个为0,一个为1
4. 将数组所有的数字分成两类,分类标准为index位是否为1
5. 分别对两组数据做累计异或得到两个值分别为a和b

```
    public void findOddTimes2Number(int[] arr) {
        //1.所有数据异或,得到a^b
        int ab = 0;
        for (int i = 0; i < arr.length; i++) {
            ab ^= arr[i];
        }
        System.out.println(ab);
        //2.找到最后一位为1的位置
        int rightBit1 = findRightBit1(ab);
        System.out.println(rightBit1);
        //3.根据这个位置将数组分为两组
        int a = 0;
        for (int i = 0; i < arr.length; i++) {
            if (0 == (arr[i] & rightBit1)) {
                a ^= arr[i];
            }
        }
        System.out.println(a);
        System.out.println(a^ab);
    }
```

5. 数组中一种数出现K次,其他数都出现M次

问题: 一个数组中有一种数出现了K次,其他数都出现了M次,M>1,K<M, 要求找到这个出现K次的数,时复杂度为O(N),额外空间复杂度为O(1)

分析:

1. 如果一个数出现M次,则他的每一位都应该出现M次
2. 将数组中所有数的每一位出现的次数累加,存入32位的数组arr中
3. arr中每一个index处的数如果时M倍数,则说明这位数不是我们要找的数
4. 如果index处的数不是M倍数,则这一位时我们要找的数

5. 遍历arr就可以组装出这个出现K次的数

```
public static int onlyKTimes(int[] arr, int k, int m) {
    int[] t = new int[32];
    // t[i] i位置的1出现了几个
    for (int num : arr) {
        for (int i = 0; i < 32; i++) {
            t[i] += (num >> i) & 1;
        }
    }
    int ans = 0;
    for (int i = 0; i < 32; i++) {
        if (t[i] % m != 0) {
            if (t[i] % m == k) {
                //出现k次,说明这个位置是我们要找的
                ans |= (1 << i);
            } else {
                return -1;
            }
        }
    }
    //ans为0时,需要特别判断
    if (ans == 0) {
        int count = 0;
        for (int num : arr) {
            if (num == 0) {
                count++;
            }
        }
        if (count != k) {
            return -1;
        }
    }
    return ans;
}
```