


# LeetCode\_只出现一次的数字IIII (找单身狗)

本博客记录一下我在力扣的刷题过程

如果有什么错误，欢迎指出，如果对你有帮助，请点个赞，谢谢

## 第一题：

### 136. 只出现一次的数字

已解答 

简单

 相关标签

 相关企业

Aa

给你一个 **非空** 整数数组 `nums`，除了某个元素只出现一次以外，其余每个元素均出现两次。找出那个只出现了一次的元素。

你必须设计并实现线性时间复杂度的算法来解决此问题，且该算法只使用常量额外空间。

CSDN @数某

编辑

这道题只需要把所有数异或起来就行了。

```
int singleNumber1(int* nums, int numsSize)
{
    int root = 0;
    for (int i = 0; i < numsSize; i++)
    {
        root ^= nums[i];
    }
    return root;
}
```

## 第二题：

# 137. 只出现一次的数字 II

已解答 

中等

 相关标签 相关企业

Aa

给你一个整数数组 `nums`，除某个元素仅出现一次外，其余每个元素都恰出现三次。请你找出并返回那个只出现了一次的元素。

你必须设计并实现线性时间复杂度的算法且使用常数级空间来解决此问题。

CSDN @数某

编辑

```
int singleNumber2(int* nums, int numsSize)
{
    int sum1 = 0;
    for (int i = 0; i < numsSize; i++)
    {
        sum1 += nums[i];
    }
    for (int i = 0; i < numsSize; i++)
    {
        for (int j = i+1; j < numsSize; j++)
        {
            if (nums[i] == nums[j])
            {
                for (int n = j; n < numsSize; n++)
                {
                    nums[n] = nums[n + 1];
                }
                numsSize--;
            }
        }
    }
    int sum2 = 0;
    for (int i = 0; i < numsSize; i++)
    {
        sum2 += nums[i];
    }
    return (sum2 * 3 - sum1)/2;
}

/*****
int singleNumber2(int* nums, int numsSize)
{
    int ans = 0;
    for (int i = 0; i < 32; ++i)
    {
        int total = 0;
        for (int j = 0; j < numsSize; ++j)
        {
            total += ((nums[j] >> i) & 1);
        }
    }
}
```


```
        if (total % 3)
        {
            ans |= (1u << i);
        }
    }
    return ans;
}
```

这道题我用了两种方法。第一种是将原来的数组消重后相加乘三减去消重之前数组的和，之后返回结果的二分之一。


第二种方法相当于是把二进制变为三进制，模上三不为0的就是结果。

## 第三题：

### 260. 只出现一次的数字 III

已解答 

中等

 相关标签

 相关企业

Aa

给你一个整数数组 `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。

你必须设计并实现线性时间复杂度的算法且仅使用常量额外空间来解决此问题。

CSDN @数某

编辑

```
int* singleNumber3(int* nums, int numsSize, int* returnSize)
{
    int i;
    int sum = 0;
    for (i = 0; i < numsSize; i++)
    {
        sum ^= nums[i];
    } //先找到两个数互相异或的结果
    int pos;
    for (i = 0; i < 32; i++)
    {
        if (sum & 1 << i)
        {
            pos = i;
            break;
        }
    } //再找到有分歧的一位。在这一位上，两个数一定是一个1一个0
    //int arr[2] = { 0 };
    int* arr = calloc(2, sizeof(int));
    for (i = 0; i < numsSize; i++)//从分歧那一位入手，分为两类
```

```
{
    if (nums[i] & 1 << pos)
    {
        arr[0] ^= nums[i]; //这一位是1的, 放在数1里
    }
    else
    {
        arr[1] ^= nums[i]; //这一位是0的, 放在数2里
    }
}
return arr;
}
```

这道题的重点是把只出现一次的两个数分开, 之后再去异或, 得到两个结果存到数组里面。

---