0136. 只出现一次的数字

ITCharge 大约 2 分钟

- 标签：位运算、数组
- 难度：简单

题目链接

- 0136. 只出现一次的数字 - 力扣

题目大意

描述：给定一个非空整数数组 `nums` ， `nums` 中除了某个元素只出现一次以外，其余每个元素均出现两次。

要求：找出那个只出现了一次的元素。

说明：

- 要求不能使用额外的存储空间。

示例：

- 示例 1：

```py
输入：[2,2,1]
输出：1
```

- 示例 2：

```py
输入：[4,1,2,1,2]
输出：4
```

解题思路

思路 1：位运算

如果没有时间复杂度和空间复杂度的限制，可以使用哈希表 / 集合来存储每个元素出现的次数，如果哈希表中没有该数字，则将该数字加入集合，如果集合中有了该数字，则从集合中删除该数字，最终成对的数字都被删除了，只剩下单次出现的元素。

但是题目要求不使用额外的存储空间，就需要用到位运算中的异或运算。

> 异或运算 $\oplus$ 的三个性质：
>
> 1. 任何数和 0 做异或运算，结果仍然是原来的数，即 $a \oplus 0 = a$。
> 2. 数和其自身做异或运算，结果是 0，即 $a \oplus a = 0$。
> 3. 异或运算满足交换率和结合律：$a \oplus b \oplus a = b \oplus a \oplus a = b \oplus (a \oplus a) = b \oplus 0 = b$。

根据异或运算的性质，对 n 个数不断 异或操作，最终可得到单次出现的元素。

思路 1：代码

```py
class Solution:
    def singleNumber(self, nums: List[int]) -> int:
        if len(nums) == 1:
            return nums[0]
        ans = 0
        for i in range(len(nums)):
            ans ^= nums[i]
        return ans
```

思路 1：复杂度分析

- **时间复杂度**：$O(n)$。

- **空间复杂度**：$O(1)$。