0287. 寻找重复数

ITCharge 大约 1 分钟

- 标签：位运算、数组、双指针、二分查找
- 难度：中等

题目链接

- 0287. 寻找重复数 - 力扣

题目大意

描述：给定一个包含 $n + 1$ 个整数的数组 $nums$，里边包含的值都在 $1 \sim n$ 之间。可知至少存在一个重复的整数。

要求：假设 $nums$ 中只存在一个重复的整数，要求找出这个重复的数。

说明：

- $1 \le n \le 10^5$。
- $nums.length == n + 1$。
- $1 \le nums[i] \le n$。
- 要求使用空间复杂度为常数级 $O(1)$，时间复杂度小于 $O(n^2)$ 的解决方法。

示例：

- 示例 1：

```py
输入：nums = [1,3,4,2,2]
输出：2
```

- 示例 2：

```py
输入：nums = [3,1,3,4,2]
输出：3
```

解题思路

思路 1：二分查找

利用二分查找的思想。

1. 使用两个指针 $left$，$right$。$left$ 指向 1，$right$ 指向 n。
2. 将区间 $[1, n]$ 分为 $[left, mid]$ 和 $[mid + 1, right]$。
3. 对于中间数 mid，统计 $nums$ 中小于等于 mid 的数个数 cnt。
4. 如果 $cnt \leq mid$，则重复数一定不会出现在左侧区间，那么从右侧区间开始搜索。
5. 如果 $cut > mid$，则重复数出现在左侧区间，则从左侧区间开始搜索。

思路 1：代码

```py
class Solution:
    def findDuplicate(self, nums: List[int]) -> int:
        n = len(nums)
        left = 1
        right = n - 1
        while left < right:
            mid = left + (right - left) // 2
            cnt = 0
            for num in nums:
                if num <= mid:
                    cnt += 1

            if cnt <= mid:
                left = mid + 1
            else:
                right = mid

        return left
```

思路 1：复杂度分析

- **时间复杂度**：$O(n \times \log n)$。
- **空间复杂度**：$O(1)$。