0080. 删除有序数组中的重复项 II

ITCharge 大约 2 分钟

- 标签：数组、双指针
- 难度：中等

题目链接

- 0080. 删除有序数组中的重复项 II - 力扣

题目大意

描述：给定一个有序数组 $nums$。

要求：在原数组空间基础上删除重复出现 2 次以上的元素，并返回删除后数组的新长度。

说明：

- $1 \leq nums.length \leq 3 * 10^4$。
- $-10^4 \leq nums[i] \leq 10^4$。
- $nums$ 已按升序排列。

示例：

- 示例 1：

```py
输入：nums = [1,1,1,2,2,3]
输出：5, nums = [1,1,2,2,3]
解释：函数应返回新长度 length = 5，并且原数组的前五个元素被修改为 1, 1, 2, 2, 3 。
不需要考虑数组中超出新长度后面的元素。
```

- 示例 2：

```py
输入：nums = [0,0,1,1,1,1,2,3,3]
输出：7, nums = [0,0,1,1,2,3,3]
解释：函数应返回新长度 length = 7，并且原数组的前五个元素被修改为 0, 0, 1, 1, 2, 3,
3 。 不需要考虑数组中超出新长度后面的元素。
```

解题思路

思路 1：快慢指针

因为数组是有序的，所以重复元素必定是连续的。可以使用快慢指针来解决。具体做法如下：

1. 使用两个指针 $slow$，$fast$。$slow$ 指针指向即将放置元素的位置，$fast$ 指针指向当前待处理元素。
2. 本题要求相同元素最多出现 2 次，并且 $slow - 2$ 是上上次放置了元素的位置。则应该检查 $nums[slow - 2]$ 和当前待处理元素 $nums[fast]$ 是否相同。
   1. 如果 $nums[slow - 2] == nums[fast]$ 时，此时必有 $nums[slow - 2] == nums[slow - 1] == nums[fast]$，则当前 $nums[fast]$ 不保留，直接向右移动快指针 $fast$。
   2. 如果 $nums[slow - 2] \neq nums[fast]$ 时，则保留 $nums[fast]$。将 $nums[fast]$ 赋值给 $nums[slow]$，同时将 $slow$ 右移。然后再向右移动快指针 $fast$。
3. 这样 $slow$ 指针左边均为处理好的数组元素，而从 $slow$ 指针指向的位置开始，$fast$ 指针左边都为舍弃的重复元素。
4. 遍历结束之后，此时 $slow$ 就是新数组的长度。

思路 1：代码

py

```
class Solution:
    def removeDuplicates(self, nums: List[int]) -> int:
        size = len(nums)
        if size <= 2:
            return size
        slow, fast = 2, 2
        while (fast < size):
            if nums[slow - 2] != nums[fast]:
                nums[slow] = nums[fast]
                slow += 1
            fast += 1
        return slow
```

思路 1：复杂度分析

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