0026. 删除有序数组中的重复项

ITCharge 大约 2 分钟

- 标签：数组、双指针
- 难度：简单

题目链接

- 0026. 删除有序数组中的重复项 - 力扣

题目大意

描述：给定一个有序数组 `nums` 。

要求：删除数组 `nums` 中的重复元素，使每个元素只出现一次。并输出去除重复元素之后数组的长度。

说明：

- 不能使用额外的数组空间，在原地修改数组，并在使用 $O(1)$ 额外空间的条件下完成。

示例：

- 示例 1：

```py
输入：nums = [1,1,2]
输出：2, nums = [1,2,_]
解释：函数应该返回新的长度 2 ，并且原数组 nums 的前两个元素被修改为 1, 2 。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 = [0,0,1,1,1,2,2,3,3,4]
输出：5, nums = [0,1,2,3,4]
解释：函数应该返回新的长度 5 ， 并且原数组 nums 的前五个元素被修改为 0, 1, 2, 3, 4 。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，那么重复的元素一定会相邻。

删除重复元素，实际上就是将不重复的元素移到数组左侧。考虑使用双指针。具体算法如下：

1. 定义两个快慢指针 `slow` ， `fast` 。其中 `slow` 指向去除重复元素后的数组的末尾位置。 `fast` 指向当前元素。
2. 令 `slow` 在后， `fast` 在前。令 `slow = 0` ， `fast = 1` 。
3. 比较 `slow` 位置上元素值和 `fast` 位置上元素值是否相等。
   - 如果不相等，则将 `slow` 后移一位，将 `fast` 指向位置的元素复制到 `slow` 位置上。
4. 将 `fast` 右移 `1` 位。
5. 重复上述 3 ~ 4 步，直到 `fast` 等于数组长度。
6. 返回 `slow + 1` 即为新数组长度。

思路 1：代码

```py
class Solution:
    def removeDuplicates(self, nums: List[int]) -> int:
        if len(nums) <= 1:
            return len(nums)

        slow, fast = 0, 1

        while (fast < len(nums)):
            if nums[slow] != nums[fast]:
                slow += 1
                nums[slow] = nums[fast]
            fast += 1

        return slow + 1
```

思路 1：复杂度分析

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