0075. 颜色分类

ITCharge 大约 2 分钟

- 标签：数组、双指针、排序
- 难度：中等

题目链接

- 0075. 颜色分类 - 力扣

题目大意

描述：给定一个数组 $nums$，元素值只有 0、1、2，分别代表红色、白色、蓝色。

要求：将数组进行排序，使得红色在前，白色在中间，蓝色在最后。

说明：

- 要求不使用标准库函数，同时仅用 [illegible] 空间，一趟扫描解决。
- $n == nums.length$。
- $1 \leq n \leq 300$。
- $nums[i]$ 为 0、1 或 2。

示例：

- 示例 1：

```py
输入：nums = [2,0,2,1,1,0]
输出：[0,0,1,1,2,2]
```

- 示例 2：

```py
输入：nums = [2,0,1]
输出：[0,1,2]
```

解题思路

思路 1：双指针 + 快速排序思想

快速排序算法中的 $partition$ 过程，利用双指针，将序列中比基准数 $pivot$ 大的元素移动到了基准数右侧，将比基准数 $pivot$ 小的元素移动到了基准数左侧。从而将序列分为了三部分：比基准数小的部分、基准数、比基准数大的部分。

这道题我们也可以借鉴快速排序算法中的 $partition$ 过程，将 1 作为基准数 $pivot$，然后将序列分为三部分：0（即比 1 小的部分）、等于 1 的部分、2（即比 1 大的部分）。具体步骤如下：

1. 使用两个指针 $left$、$right$，分别指向数组的头尾。$left$ 表示当前处理好红色元素的尾部，$right$ 表示当前处理好蓝色的头部。
2. 再使用一个下标 $index$ 遍历数组，如果遇到 $nums[index] == 0$，就交换 $nums[index]$ 和 $nums[left]$，同时将 $left$ 右移。如果遇到 $nums[index] == 2$，就交换 $nums[index]$ 和 $nums[right]$，同时将 $right$ 左移。
3. 直到 $index$ 移动到 $right$ 位置之后，停止遍历。遍历结束之后，此时 $left$ 左侧都是红色，$right$ 右侧都是蓝色。

注意：移动的时候需要判断 $index$ 和 $left$ 的位置，因为 $left$ 左侧是已经处理好的数组，所以需要判断 $index$ 的位置是否小于 $left$，小于的话，需要更新 $index$ 位置。

思路 1：代码

```py
class Solution:
    def sortColors(self, nums: List[int]) -> None:
        left = 0
        right = len(nums) - 1
        index = 0
        while index <= right:
            if index < left:
                index += 1
            elif nums[index] == 0:
                nums[index], nums[left] = nums[left], nums[index]
                left += 1
```

```
        elif nums[index] == 2:
            nums[index], nums[right] = nums[right], nums[index]
            right -= 1
        else:
            index += 1
```

思路 1：复杂度分析

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