0189. 轮转数组

ITCharge 大约 2 分钟

- 标签：数组、数学、双指针
- 难度：中等

题目链接

- 0189. 轮转数组 - 力扣

题目大意

描述：给定一个数组 $nums$，再给定一个数字 k。

要求：将数组中的元素向右移动 k 个位置。

说明：

- $1 \leq nums.length \leq 10^5$。
- $-2^{31} \leq nums[i] \leq 2^{31} - 1$。
- $0 \leq k \leq 10^5$。
- 使用空间复杂度为 $O(1)$ 的原地算法解决这个问题。

示例：

- 示例 1：

```py
输入：nums = [1,2,3,4,5,6,7], k = 3
输出：[5,6,7,1,2,3,4]
解释：
向右轮转 1 步：[7,1,2,3,4,5,6]
向右轮转 2 步：[6,7,1,2,3,4,5]
向右轮转 3 步：[5,6,7,1,2,3,4]
```

- 示例 2：

```py
输入：nums = [-1,-100,3,99], k = 2
输出：[3,99,-1,-100]
解释：
向右轮转 1 步：[99,-1,-100,3]
向右轮转 2 步：[3,99,-1,-100]
```

解题思路

思路 1： 数组翻转

可以用一个新数组，先保存原数组的后 k 个元素，再保存原数组的前 $n-k$ 个元素。但题目要求不使用额外的数组空间，那么就需要在原数组上做操作。

我们可以先把整个数组翻转一下，这样后半段元素就到了前边，前半段元素就到了后边，只不过元素顺序是反着的。我们再从 k 位置分隔开，将 $[0...k-1]$ 区间上的元素和 $[k...n-1]$ 区间上的元素再翻转一下，就得到了最终结果。

具体步骤：

1. 将数组 $[0, n-1]$ 位置上的元素全部翻转。
2. 将数组 $[0, k-1]$ 位置上的元素进行翻转。
3. 将数组 $[k, n-1]$ 位置上的元素进行翻转。

思路 1：代码

```py
class Solution:
    def rotate(self, nums: List[int], k: int) -> None:
        n = len(nums)
        k = k % n
        self.reverse(nums, 0, n-1)
        self.reverse(nums, 0, k-1)
        self.reverse(nums, k, n-1)
    def reverse(self, nums: List[int], left: int, right: int) -> None:
        while left < right :
```

```
            tmp = nums[left]
            nums[left] = nums[right]
            nums[right] = tmp
            left += 1
            right -= 1
```

思路 1：复杂度分析

- **时间复杂度**：$O(n)$。翻转的时间复杂度为 $O(n)$ 。
- **空间复杂度**：$O(1)$。