0015. 三数之和

ITCharge 大约 2 分钟

- 标签：数组、双指针、排序
- 难度：中等

题目链接

- 0015. 三数之和 - 力扣

题目大意

描述：给定一个整数数组 $nums$。

要求：判断 $nums$ 中是否存在三个元素 a、b、c，满足 $a + b + c == 0$。要求找出所有满足要求的不重复的三元组。

说明：

- $3 \leq nums.length \leq 3000$。
- $-10^5 \leq nums[i] \leq 10^5$。

示例：

- 示例 1：

```py
输入：nums = [-1,0,1,2,-1,-4]
输出：[[-1,-1,2],[-1,0,1]]
```

- 示例 2：

```py
输入：nums = [0,1,1]
输出：[]
```

解题思路

思路 1：对撞指针

直接三重遍历查找 a、b、c 的时间复杂度是：$O(n^3)$。我们可以通过一些操作来降低复杂度。

先将数组进行排序，以保证按顺序查找 a、b、c 时，元素值为升序，从而保证所找到的三个元素是不重复的。同时也方便下一步使用双指针减少一重遍历。时间复杂度为：$O(n \times \log n)$。

第一重循环遍历 a，对于每个 a 元素，从 a 元素的下一个位置开始，使用对撞指针 $left$，$right$。$left$ 指向 a 元素的下一个位置，$right$ 指向末尾位置。先将 $left$ 右移、$right$ 左移去除重复元素，再进行下边的判断。

1. 如果 $nums[a] + nums[left] + nums[right] == 0$，则得到一个解，将其加入答案数组中，并继续将 $left$ 右移，$right$ 左移；
2. 如果 $nums[a] + nums[left] + nums[right] > 0$，说明 $nums[right]$ 值太大，将 $right$ 向左移；
3. 如果 $nums[a] + nums[left] + nums[right] < 0$，说明 $nums[left]$ 值太小，将 $left$ 右移。

思路 1：代码

```py
class Solution:
    def threeSum(self, nums: List[int]) -> List[List[int]]:
        n = len(nums)
        nums.sort()
        ans = []

        for i in range(n):
            if i > 0 and nums[i] == nums[i - 1]:
                continue
            left = i + 1
            right = n - 1
            while left < right:
                while left < right and left > i + 1 and nums[left] == nums[left
```

```
- 1]:
                    left += 1
                while left < right and right < n - 1 and nums[right + 1] ==
nums[right]:
                    right -= 1
                if left < right and nums[i] + nums[left] + nums[right] == 0:
                    ans.append([nums[i], nums[left], nums[right]])
                    left += 1
                    right -= 1
                elif nums[i] + nums[left] + nums[right] > 0:
                    right -= 1
                else:
                    left += 1
        return ans
```

思路 1：复杂度分析

- **时间复杂度**：$O(n^2)$。
- **空间复杂度**：$O(n)$。