# 题目

给定一个包含非负整数的数组，你的任务是统计其中可以组成三角形三条边的三元组个数。

示例 1:

输入: [2,2,3,4]

输出: 3

解释:

有效的组合是:

2,3,4 (使用第一个 2)

2,3,4 (使用第二个 2)

2,2,3

**注意：**

数组长度不超过1000。

数组里整数的范围为 [0, 1000]。

# 分析

思路：这是典型的基数排序。

## 方法一：排序+二分查找

## 方法二：排序+双指针

class Solution {

public:

int triangleNumber(vector<int>& nums) {

int n = nums.size();

sort(nums.begin(), nums.end());

int ans = 0;

for (int i = 0; i < n - 2; ++i) { // 第一个边的索引 i，最多到倒数第三个数

int k = i + 2; // 第三条边的索引 k 从 i+2 开始，因为要保证三角形成立

for (int j = i + 1; j < n - 1 && nums[i] != 0; ++j) { // 第二个边的索引 j，最多到倒数第二个数，且 nums[i] != 0

// 在有序数组中找到第一个大于等于 nums[i] + nums[j] 的数的索引

while (k < n && nums[i] + nums[j] > nums[k]) {

++k;

}

// 如果 k > j，则说明存在三角形，此时组合数为 k - j - 1

ans += k - j - 1;

}

}

return ans;

}

};

复杂度分析

时间复杂度： O(n^2)，其中n是数组nums的长度。我们需要O(nlogn)的时间对数组nums进行排序，随后需要O(n^2)的时间使用一重循环枚举a的下标以及使用双指针维护b,c的下标。

空间复杂度：O(logn)，即为排序需要的栈空间。