

15. 3Sum



Given an array `nums` of n integers, are there elements a, b, c in `nums` such that $a + b + c = 0$? Find all unique triplets in the array which gives the sum of zero.

Notice that the solution set must not contain duplicate triplets.

Example 1:

Input: `nums = [-1,0,1,2,-1,-4]`
Output: `[[-1,-1,2],[-1,0,1]]`

Example 2:

Input: `nums = []`
Output: `[]`

Example 3:

Input: `nums = [0]`
Output: `[]`

Constraints:

- $0 \leq \text{nums.length} \leq 3000$
- $-10^5 \leq \text{nums}[i] \leq 10^5$

三数之和等于0

Two pointers, Array

1. **sort** the array
2. iterate the array **[0, length-2)** as at least two more numbers after it, hold a fixed number and two pointers in **opposite direction** the same approach as 2sum problem
3. **skip the duplicate number by comparing with the number before or after it**

16. 3Sum Closest



Given an array `nums` of n integers and an integer `target`, find three integers in `nums` such that the sum is closest to `target`. Return the sum of the three integers. You may assume that each input would have exactly one solution.

Example 1:

Input: `nums = [-1,2,1,-4]`, `target = 1`

Output: `2`

Explanation: The sum that is closest to the target is 2. $(-1 + 2 + 1 = 2)$.

Constraints:

- $3 \leq \text{nums.length} \leq 10^3$
- $-10^3 \leq \text{nums}[i] \leq 10^3$
- $-10^4 \leq \text{target} \leq 10^4$

求三数之和最接近给定值

Two pointers, Array

1. first find the raw sum of the first three numbers, and sort the array
2. the same approach as **15. 3sum**
3. update the result by comparing the absolute difference between (3sum, target) and (result, target)

18. 4Sum

Given an array `nums` of n integers and an integer `target`, are there elements a , b , c , and d in `nums` such that $a + b + c + d = \text{target}$? Find all unique quadruplets in the array which gives the sum of `target`.

Notice that the solution set must not contain duplicate quadruplets.

Example 1:

Input: `nums = [1,0,-1,0,-2,2]`, `target = 0`

Output: `[[-2,-1,1,2],[-2,0,0,2],[-1,0,0,1]]`

Example 2:

Input: nums = [], target = 0
Output: []

Constraints:

- $0 \leq \text{nums.length} \leq 200$
- $-10^9 \leq \text{nums}[i] \leq 10^9$
- $-10^9 \leq \text{target} \leq 10^9$

四数之和等于给定值

Two pointers, Array

1. sort the array, same approach as **15. 3sum**
2. two for loops to hold two fixed values, and the same two pointers approach in opposite direction to compare the sum of two pointers with the **target - sum of two fixed values**
3. **注意去重**

611. Valid Triangle Number



Given an array consists of non-negative integers, your task is to count the number of triplets chosen from the array that can make triangles if we take them as side lengths of a triangle.

Example 1:

Input: [2,2,3,4]

Output: 3

Explanation:

Valid combinations are:

2,3,4 (using the first 2)

2,3,4 (using the second 2)

2,2,3

Note:

1. The length of the given array won't exceed 1000.
2. The integers in the given array are in the range of [0, 1000].

有多少有效的三角形

Two pointers, Array

1. sort array, iterate it backwards [**length-1, 2**] as at least two numbers before it
 2. two pointers, compare the sum of two sides with the third side, if the sum is greater, **count+=right-left**, move left and right until they meet
-