Problem 18: 4Sum

Problem Information

Difficulty: Medium
Acceptance Rate: 39.35%
Paid Only: No
Tags: Array, Two Pointers, Sorting

Problem Description

Given an array `nums` of `n` integers, return _an array of all the**unique** quadruplets_ `[nums[a], nums[b], nums[c], nums[d]]` such that:

* `0 <= a, b, c, d < n` * `a`, `b`, `c`, and `d` are **distinct**. * `nums[a] + nums[b] + nums[c] + nums[d] == target`

You may return the answer in **any order**.

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 = [2,2,2,2,2], target = 8 **Output:** [[2,2,2,2]]

Constraints:

* `1 <= nums.length <= 200` * `-109 <= nums[i] <= 109` * `-109 <= target <= 109`

Code Snippets

C++:

```
class Solution {
public:
vector<vector<int>> fourSum(vector<int>& nums, int target) {

}
};
```

Java:

```
class Solution {
public List<List<Integer>> fourSum(int[] nums, int target) {

}
}
```

Python3:

```
class Solution:
def fourSum(self, nums: List[int], target: int) -> List[List[int]]:
```