# 题目

设计一个算法，找出数组中两数之和为指定值的所有整数对。一个数只能属于一个数对。

示例 1:

输入: nums = [5,6,5], target = 11

输出: [[5,6]]

示例 2:

输入: nums = [5,6,5,6], target = 11

输出: [[5,6],[5,6]]

提示：

nums.length <= 100000

-10^5 <= nums[i], target <= 10^5

# 分析

## 方法一：哈希表

你可以使用哈希表来解决这个问题。遍历数组中的每个元素，对于每个元素，判断target - nums[i]是否在哈希表中，如果在，则找到了一对数之和为target的数对。

具体步骤如下：

1、定义一个哈希表hash\_map<int, int>，用来存储每个元素出现的次数。

2、遍历数组nums，对于每个元素num，判断target - num是否在哈希表中：

- 如果在，将[num, target - num]加入结果集中，并更新哈希表中target - num的次数减一。

- 如果不在，将num的次数加一。

3、返回结果集。

下面是具体的实现：

class Solution {

public:

vector<vector<int>> pairSums(vector<int>& nums, int target) {

unordered\_map<int, int> num\_counts;

vector<vector<int>> result;

for (int num : nums) {

if (num\_counts[target - num] > 0) {

result.push\_back({num, target - num});

num\_counts[target - num]--;

} else {

num\_counts[num]++;

}

}

return result;

}

};

在这个实现中，我们使用哈希表num\_counts来记录每个元素出现的次数。对于每个元素num，如果target - num在哈希表中，并且次数大于零，说明找到了一对数之和为target的数对，将其加入结果集中，并更新哈希表中target - num的次数减一。如果不在哈希表中，则将num的次数加一。最终返回结果集即为所有满足条件的数对。