# 题目

给你两个下标从 0 开始的整数数组 nums1 和 nums2 ，请你返回一个长度为 2 的列表 answer ，其中：

answer[0] 是 nums1 中所有 不 存在于 nums2 中的 不同 整数组成的列表。

answer[1] 是 nums2 中所有 不 存在于 nums1 中的 不同 整数组成的列表。

注意：列表中的整数可以按 任意 顺序返回。

示例 1：

输入：nums1 = [1,2,3], nums2 = [2,4,6]

输出：[[1,3],[4,6]]

解释：

对于 nums1 ，nums1[1] = 2 出现在 nums2 中下标 0 处，然而 nums1[0] = 1 和 nums1[2] = 3 没有出现在 nums2 中。因此，answer[0] = [1,3]。

对于 nums2 ，nums2[0] = 2 出现在 nums1 中下标 1 处，然而 nums2[1] = 4 和 nums2[2] = 6 没有出现在 nums2 中。因此，answer[1] = [4,6]。

示例 2：

输入：nums1 = [1,2,3,3], nums2 = [1,1,2,2]

输出：[[3],[]]

解释：

对于 nums1 ，nums1[2] 和 nums1[3] 没有出现在 nums2 中。由于 nums1[2] == nums1[3] ，二者的值只需要在 answer[0] 中出现一次，故 answer[0] = [3]。

nums2 中的每个整数都在 nums1 中出现，因此，answer[1] = [] 。

提示：

1 <= nums1.length, nums2.length <= 1000

-1000 <= nums1[i], nums2[i] <= 1000

# 分析

## 方法一：哈希集合

思路：

首先将nums1和nums2中的元素分别插入到两个unordered\_set中，然后遍历这两个unordered\_set，找出在其中一个集合中出现而在另一个集合中未出现的元素，分别放入结果向量res的第一个和第二个元素中。最后返回结果向量res。

代码：

class Solution {

public:

vector<vector<int>> findDifference(vector<int>& nums1, vector<int>& nums2) {

unordered\_set<int> set1, set2; // nums1 与 nums2 所有元素的哈希集合

for (int num: nums1) {

set1.insert(num);

}

for (int num: nums2) {

set2.insert(num);

}

vector<vector<int>> res(2);

for (int num: set1) {

if (!set2.count(num)) {

res[0].push\_back(num);

}

}

for (int num: set2) {

if (!set1.count(num)) {

res[1].push\_back(num);

}

}

return res;

}

};