349. 两个数组的交集

力扣题目链接(opens new window)

题意：给定两个数组，编写一个函数来计算它们的交集。

示例 1：

```
输入：nums1 = [1,2,2,1], nums2 = [2,2]
输出：[2]
```

示例 2：

```
输入：nums1 = [4,9,5], nums2 = [9,4,9,8,4]
输出：[9,4]
```

说明： 输出结果中的每个元素一定是唯一的。 我们可以不考虑输出结果的顺序。

思路

这道题目，主要要学会使用一种哈希数据结构：unordered_set，这个数据结构可以解决很多类似的问题。

注意题目特意说明：**输出结果中的每个元素一定是唯一的，也就是说输出的结果的去重的， 同时可以不考虑输出结果的顺序**

这道题用暴力的解法时间复杂度是O(n^2)，那来看看使用哈希法进一步优化。

那么用数组来做哈希表也是不错的选择，例如242. 有效的字母异位词(opens new window)

但是要注意，**使用数组来做哈希的题目，是因为题目都限制了数值的大小。**

而这道题目没有限制数值的大小，就无法使用数组来做哈希表了。

而且如果哈希值比较少、特别分散、跨度非常大，使用数组就造成空间的极大浪费。

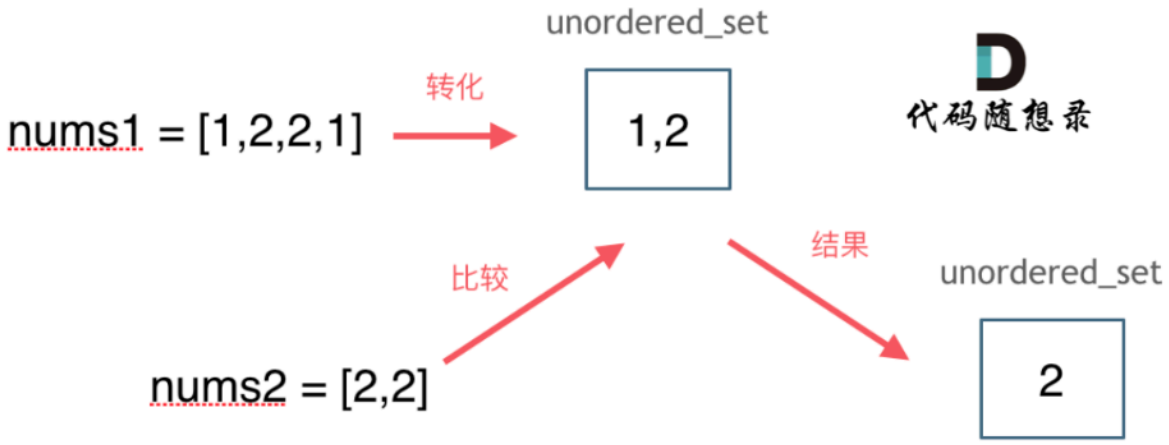
> **Note**
>
> Set在python中相关的数据结构有两种：
>
> `set` **用于动态操作，** `frozenset` **用于需要"集合本身作为不可变对象"的场景。**
>
> (1) `set` —— **可变集合**
>
> - 用 `set()` 或 `{}`（注意：`{}` 是空字典，空集合必须写成 `set()`）创建。
> - **元素唯一、无序**（Python 3.7+ 虽然底层有序，但官方仍视为无序）。
> - **可增删元素**（支持 `.add()`，`.remove()`，`.discard()` 等方法）。

- **不可哈希** → 不能作为字典的键，也不能放进另一个 set 里。

(2) `frozenset` —— **不可变集合**

- 用 `frozenset()` 创建。
- 同样元素唯一、无序。
- **不可修改**（没有 `.add()` 等方法）。
- **可哈希** → 可以作为字典的键，也可以作为另一个 set 的元素。

思路如图所示：



- 时间复杂度: O(n + m) m 是最后要把 set转成vector
- 空间复杂度: O(n)

使用集合实现：

```
class Solution:
    def intersection(self, nums1: List[int], nums2: List[int]) -> List[int]:
        return list(set(nums1) & set(nums2))
```

使用字典和集合：

```
class Solution:
    def intersection(self, nums1: List[int], nums2: List[int]) -> List[int]:
        table = {}
        for num in nums1:
            table[num] = table.get(num,0) + 1

        res = set()
        for num in nums2:
            if num in table:
                res.add(num)
                del table[num]

        return list(res)
```

使用数组实现：

```python
class Solution:
    def intersection(self, nums1: List[int], nums2: List[int]) -> List[int]:
        count1 = [0]*1001
        count2 = [0]*1001
        result = []

        for i in range(len(nums1)):
            count1[nums1[i]] += 1

        for i in range(len(nums2)):
            count2[nums2[i]] += 1

        for k in range(1001):
            if count1[k]*count2[k]>0:
                result.append(k)
        return result
```

> **Note**
>
> **用了什么数据结构?**
>
> - `count1` **和** `count2` 是 **Python 的列表（list）**，但在这里被当作 **“哈希表的替代品”** 使用。
> - 具体来说，这是一个 **“以数值本身作为索引”的数组**，也叫 **“桶（bucket）”** 或 **“直接寻址表（direct address table）”**。
>
> 所以：底层数据结构是 **list**，但逻辑上模拟了 **哈希表（键 → 计数）** 的功能。