599 Minimum Index Sum of Two Lists

2018年4月16日 18:45

Suppose Andy and Doris want to choose a restaurant for dinner, and they both have a list of favorite restaurants represented by strings.

You need to help them find out their **common interest** with the **least list index sum**. If there is a choice tie between answers, output all of them with no order requirement. You could assume there always exists an answer.

Example 1:

Input:

["Shogun", "Tapioca Express", "Burger King", "KFC"]

["Piatti", "The Grill at Torrey Pines", "Hungry Hunter Steakhouse", "Shogun"]

Output: ["Shogun"]

Explanation: The only restaurant they both like is "Shogun".

Example 2:

Input:

["Shogun", "Tapioca Express", "Burger King", "KFC"]

["KFC", "Shogun", "Burger King"]

Output: ["Shogun"]

Explanation: The restaurant they both like and have the least index sum is "Shogun" with index sum 1 (0+1).

Note:

1. The length of both lists will be in the range of [1, 1000].
2. The length of strings in both lists will be in the range of [1, 30].
3. The index is starting from 0 to the list length minus 1.
4. No duplicates in both lists.

来自 <https://leetcode.com/problems/minimum-index-sum-of-two-lists/description/>

假设Andy和Doris想在晚餐时选择一家餐厅，并且他们都有一个表示最喜爱餐厅的列表，每个餐厅的名字用字符串表示。

你需要帮助他们用**最少的索引和**找出他们**共同喜爱的餐厅**。 如果答案不止一个，则输出所有答案并且不考虑顺序。 你可以假设总是存在一个答案。

示例 1:

输入:

["Shogun", "Tapioca Express", "Burger King", "KFC"]

["Piatti", "The Grill at Torrey Pines", "Hungry Hunter Steakhouse", "Shogun"]

输出: ["Shogun"]

解释: 他们唯一共同喜爱的餐厅是“Shogun”。

示例 2:

输入:

["Shogun", "Tapioca Express", "Burger King", "KFC"]

["KFC", "Shogun", "Burger King"]

输出: ["Shogun"]

解释: 他们共同喜爱且具有最小索引和的餐厅是“Shogun”，它有最小的索引和1(0+1)。

提示:

1. 两个列表的长度范围都在 [1, 1000]内。
2. 两个列表中的字符串的长度将在[1，30]的范围内。
3. 下标从0开始，到列表的长度减1。
4. 两个列表都没有重复的元素。

Solution for Python3:

```python
class Solution:
    def findRestaurant(self, list1, list2):
        """
        :type list1: List[str]
        :type list2: List[str]
        :rtype: List[str]
        """
        d = {s:i for i, s in enumerate(list1)}
        small, ans =20000, []
        for j, s in enumerate(list2):
            i = d.get(s, 20000)
            if i + j < small:
                small = i + j
                ans = [s]
            elif i + j == small:
                ans.append(s)
        return ans
```

Solution for C++:

```cpp
class Solution {
public:
    vector<string> findRestaurant(vector<string>&
list1, vector<string>& list2) {
        unordered_map<string, int> m;
        vector<string> res;
```

```
            int minV = INT_MAX, t;
4           for (int i = 0; i < list1.size(); i++)
                m[list1[i]] = i;
5           for (int i = 0; i < list2.size(); i++) {
                if (m.count(list2[i]) != 0) {
6                   t = m[list2[i]] + i;
                    if (t < minV) {
7                       minV = t;
                        res.clear();
8                       res.push_back(list2[i]);
                    } else if (t == minV) {
9                       res.push_back(list2[i]);
                    }
10              }
            }
11          return res;
        }
12  };
13
14
15
16
17
18
19
20
21
22
```

23