1.2 TO FIND THE NUMBER OF ELEMENTS IN ONE ARRAY EXIST IN THE OTHER ARRAY

Question:

You are given two integer arrays nums1 and nums2 of sizes n and m, respectively. Calculate the following values: answer1 : the number of indices i such that nums1[i] exists in nums2. answer2 : the number of indices i such that nums2[i] exists in nums1 Return [answer1,answer2].

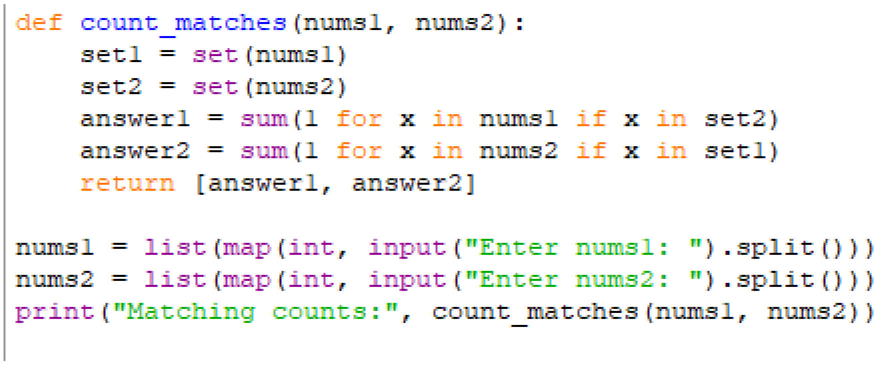
AIM:

To find how many elements of one array exist in the other array and return the counts as [answer1, answer2].

ALGORITHM:

1. Store the elements of nums1 in a set set1 for fast lookup.
2. Store the elements of nums2 in a set set2.
3. Initialize answer1 = 0, answer2 = 0.
4. Traverse each element in nums1:
   - If the element exists in set2, increment answer1.
5. Traverse each element in nums2:
   - If the element exists in set1, increment answer2.
6. Return [answer1, answer2].

PROGRAM:



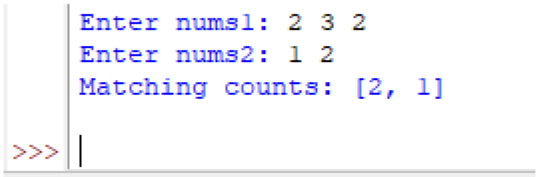
```
def count_matches(nums1, nums2):
    set1 = set(nums1)
    set2 = set(nums2)
    answer1 = sum(1 for x in nums1 if x in set2)
    answer2 = sum(1 for x in nums2 if x in set1)
    return [answer1, answer2]

nums1 = list(map(int, input("Enter nums1: ").split()))
nums2 = list(map(int, input("Enter nums2: ").split()))
print("Matching counts:", count_matches(nums1, nums2))
```

Input:

nums1 = [2,3,2]

nums2 = [1,2]

Output:



```
Enter nums1: 2 3 2
Enter nums2: 1 2
Matching counts: [2, 1]

>>> |
```

RESULT:

Thus the program is successfully executed, and the output is verified.

PERFORMANCE ANALYSIS:

- Time Complexity:
  - Creating sets → O(n + m)
  - Checking membership → O(n + m)
  - Total: O(n + m)
- Space Complexity:
  - Two sets storing up to n + m elements → O(n + m)