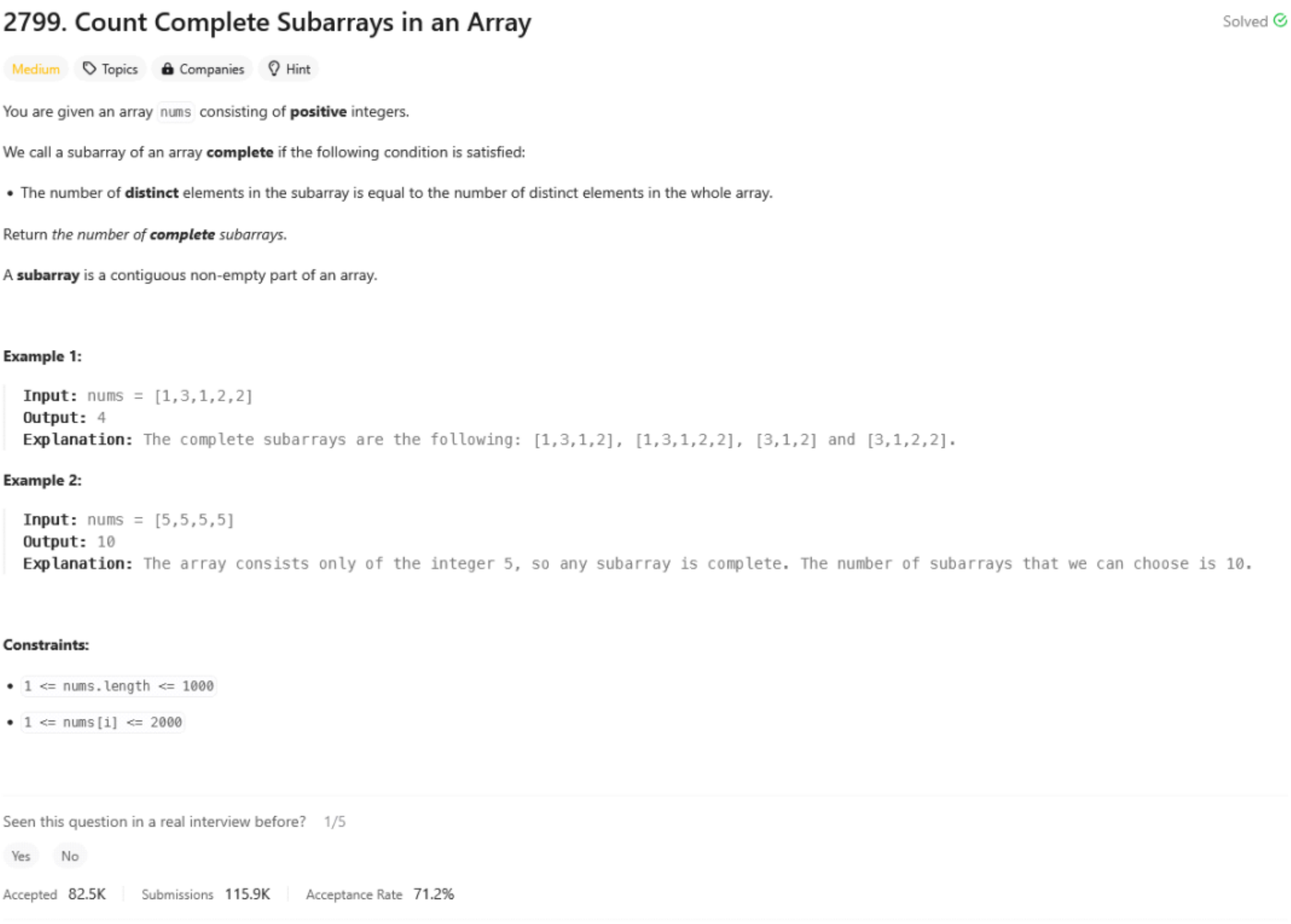
Thursday, April 24, 2025 4:38 PM

Count Complete Subarrays in an Array



2799. Count Complete Subarrays in an Array

Solved ✓

Medium Topics Companies Hint

You are given an array `nums` consisting of **positive** integers.

We call a subarray of an array **complete** if the following condition is satisfied:

- The number of **distinct** elements in the subarray is equal to the number of distinct elements in the whole array.

Return *the number of* ***complete*** *subarrays*.

A **subarray** is a contiguous non-empty part of an array.

Example 1:

```
Input: nums = [1,3,1,2,2]
Output: 4
Explanation: The complete subarrays are the following: [1,3,1,2], [1,3,1,2,2], [3,1,2] and [3,1,2,2].
```

Example 2:

```
Input: nums = [5,5,5,5]
Output: 10
Explanation: The array consists only of the integer 5, so any subarray is complete. The number of subarrays that we can choose is 10.
```

Constraints:

- `1 <= nums.length <= 1000`
- `1 <= nums[i] <= 2000`

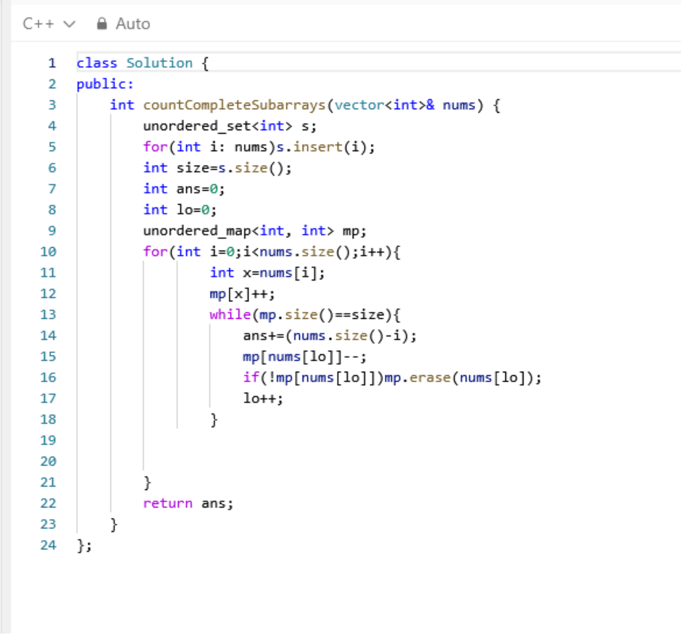
Seen this question in a real interview before? 1/5

Yes No

Accepted 82.5K | Submissions 115.9K | Acceptance Rate 71.2%

```
class Solution{
 public:
   int countCompleteSubarrays(vector<int>&nums){
       unordered_set<int>s;
     for(int i:nums) s.insert(i);
      int size = s.size();
      int ans = 0;
      int lo = 0;
int ans = 0;
int lo = 0;
unordered_map<int, int> mp;
for(int i=0; i<nums.size(); i++){
        int x = nums[i];
        mp[x]++;
        while(mp.size() == size){
          ans+= (nums.size()-i);
          mp[nums[lo]]--;
          if(!mp[nums[lo]]) mp.erase(nums[lo]);
          lo++; }}
        return ans; }}.
```

T.C. O(n) S.C. O(n)



```cpp
class Solution {
public:
    int countCompleteSubarrays(vector<int>& nums) {
        unordered_set<int> s;
        for(int i: nums)s.insert(i);
        int size=s.size();
        int ans=0;
        int lo=0;
        unordered_map<int, int> mp;
        for(int i=0;i<nums.size();i++){
                int x=nums[i];
                mp[x]++;
                while(mp.size()==size){
                    ans+=(nums.size()-i);
                    mp[nums[lo]]--;
                    if(!mp[nums[lo]])mp.erase(nums[lo]);
                    lo++;
                }


        }
        return ans;
    }
};
```