Problem 3748: Count Stable Subarrays

Problem Information

Difficulty: Hard
Acceptance Rate: 30.57%
Paid Only: No
Tags: Array, Binary Search, Prefix Sum

Problem Description

You are given an integer array `nums`.

A **subarray** of `nums` is called **stable** if it contains **no inversions** , i.e., there is no pair of indices `i < j` such that `nums[i] > nums[j]`.

You are also given a **2D integer array** `queries` of length `q`, where each `queries[i] = [li, ri]` represents a query. For each query `[li, ri]`, compute the number of **stable subarrays** that lie entirely within the segment `nums[li..ri]`.

Return an integer array `ans` of length `q`, where `ans[i]` is the answer to the `ith` query.

Note :

* A single element subarray is considered stable.

Example 1:

Input: nums = [3,1,2], queries = [[0,1],[1,2],[0,2]]

Output: [2,3,4]

Explanation:

* For `queries[0] = [0, 1]`, the subarray is `[nums[0], nums[1]] = [3, 1]`. * The stable subarrays are `[3]` and `[1]`. The total number of stable subarrays is 2. * For `queries[1] = [1, 2]`, the subarray is `[nums[1], nums[2]] = [1, 2]`. * The stable subarrays are `[1]`, `[2]`, and `[1, 2]`.

The total number of stable subarrays is 3. * For `queries[2] = [0, 2]`, the subarray is `[nums[0], nums[1], nums[2]] = [3, 1, 2]`. * The stable subarrays are `[3]`, `[1]`, `[2]`, and `[1, 2]`. The total number of stable subarrays is 4.

Thus, `ans = [2, 3, 4]`.

Example 2:

Input: nums = [2,2], queries = [[0,1],[0,0]]

Output: [3,1]

Explanation:

* For `queries[0] = [0, 1]`, the subarray is `[nums[0], nums[1]] = [2, 2]`. * The stable subarrays are `[2]`, `[2]`, and `[2, 2]`. The total number of stable subarrays is 3. * For `queries[1] = [0, 0]`, the subarray is `[nums[0]] = [2]`. * The stable subarray is `[2]`. The total number of stable subarrays is 1.

Thus, `ans = [3, 1]`.

Constraints:

* `1 <= nums.length <= 105` * `1 <= nums[i] <= 105` * `1 <= queries.length <= 105` * `queries[i] = [li, ri]` * `0 <= li <= ri <= nums.length - 1`

Code Snippets

C++:

```
class Solution {
public:
vector<long long> countStableSubarrays(vector<int>& nums,
vector<vector<int>>& queries) {

}
};
```

Java:

```
class Solution {
public long[] countStableSubarrays(int[] nums, int[][] queries) {

}
}
```

Python3:

```
class Solution:
def countStableSubarrays(self, nums: List[int], queries: List[List[int]]) ->
List[int]:
```