Problem 2393: Count Strictly Increasing Subarrays

Problem Information

Difficulty: Medium
Acceptance Rate: 71.25%
Paid Only: Yes
Tags: Array, Math, Dynamic Programming

Problem Description

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

Return _the number of**subarrays** of _`nums` _that are in**strictly increasing** order._

A **subarray** is a **contiguous** part of an array.

Example 1:

Input: nums = [1,3,5,4,4,6] **Output:** 10 **Explanation:** The strictly increasing subarrays are the following: - Subarrays of length 1: [1], [3], [5], [4], [4], [6]. - Subarrays of length 2: [1,3], [3,5], [4,6]. - Subarrays of length 3: [1,3,5]. The total number of subarrays is 6 + 3 + 1 = 10.

Example 2:

Input: nums = [1,2,3,4,5] **Output:** 15 **Explanation:** Every subarray is strictly increasing. There are 15 possible subarrays that we can take.

Constraints:

* `1 <= nums.length <= 105` * `1 <= nums[i] <= 106`

Code Snippets

C++:

```
class Solution {
public:
long long countSubarrays(vector<int>& nums) {

}
};
```

Java:

```
class Solution {
public long countSubarrays(int[] nums) {

}
}
```

Python3:

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