Problem 2104: Sum of Subarray Ranges

Problem Information

Difficulty: Medium
Acceptance Rate: 60.35%
Paid Only: No
Tags: Array, Stack, Monotonic Stack

Problem Description

You are given an integer array `nums`. The **range** of a subarray of `nums` is the difference between the largest and smallest element in the subarray.

Return _the**sum of all** subarray ranges of _`nums` _._

A subarray is a contiguous **non-empty** sequence of elements within an array.

Example 1:

Input: nums = [1,2,3] **Output:** 4 **Explanation:** The 6 subarrays of nums are the following: [1], range = largest - smallest = 1 - 1 = 0 [2], range = 2 - 2 = 0 [3], range = 3 - 3 = 0 [1,2], range = 2 - 1 = 1 [2,3], range = 3 - 2 = 1 [1,2,3], range = 3 - 1 = 2 So the sum of all ranges is 0 + 0 + 0 + 1 + 1 + 2 = 4.

Example 2:

Input: nums = [1,3,3] **Output:** 4 **Explanation:** The 6 subarrays of nums are the following: [1], range = largest - smallest = 1 - 1 = 0 [3], range = 3 - 3 = 0 [3], range = 3 - 3 = 0 [1,3], range = 3 - 1 = 2 [3,3], range = 3 - 3 = 0 [1,3,3], range = 3 - 1 = 2 So the sum of all ranges is 0 + 0 + 0 + 2 + 0 + 2 = 4.

Example 3:

Input: nums = [4,-2,-3,4,1] **Output:** 59 **Explanation:** The sum of all subarray ranges of nums is 59.

Constraints:

* `1 <= nums.length <= 1000` * `-109 <= nums[i] <= 109`

Follow-up: Could you find a solution with `O(n)` time complexity?

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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