

Problem 581: Shortest Unsorted Continuous Subarray

Problem Information

Difficulty: Medium

Acceptance Rate: 37.77%

Paid Only: No

Tags: Array, Two Pointers, Stack, Greedy, Sorting, Monotonic Stack

Problem Description

Given an integer array `nums`, you need to find one **continuous subarray** such that if you only sort this subarray in non-decreasing order, then the whole array will be sorted in non-decreasing order.

Return `_the shortest such subarray and output its length_`.

Example 1:

Input: `nums = [2,6,4,8,10,9,15]` **Output:** `5` **Explanation:** You need to sort `[6, 4, 8, 10, 9]` in ascending order to make the whole array sorted in ascending order.

Example 2:

Input: `nums = [1,2,3,4]` **Output:** `0`

Example 3:

Input: `nums = [1]` **Output:** `0`

Constraints:

`1 <= nums.length <= 104` `-105 <= nums[i] <= 105`

Follow up: Can you solve it in `O(n)` time complexity?

Code Snippets

C++:

```
class Solution {  
public:  
    int findUnsortedSubarray(vector<int>& nums) {  
  
    }  
};
```

Java:

```
class Solution {  
    public int findUnsortedSubarray(int[] nums) {  
  
    }  
}
```

Python3:

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