Problem 3523: Make Array Non-decreasing

Problem Information

Difficulty: Medium
Acceptance Rate: 56.23%
Paid Only: No
Tags: Array, Stack, Greedy, Monotonic Stack

Problem Description

You are given an integer array `nums`. In one operation, you can select a subarray and replace it with a single element equal to its **maximum** value.

Return the **maximum possible size** of the array after performing zero or more operations such that the resulting array is **non-decreasing**.

Example 1:

Input: nums = [4,2,5,3,5]

Output: 3

Explanation:

One way to achieve the maximum size is:

1. Replace subarray `nums[1..2] = [2, 5]` with `5` -> `[4, 5, 3, 5]`. 2. Replace subarray `nums[2..3] = [3, 5]` with `5` -> `[4, 5, 5]`.

The final array `[4, 5, 5]` is non-decreasing with size 3.

Example 2:

Input: nums = [1,2,3]

Output: 3

Explanation:

No operation is needed as the array `[1,2,3]` is already non-decreasing.

Constraints:

* `1 <= nums.length <= 2 * 105` * `1 <= nums[i] <= 2 * 105`

Code Snippets

C++:

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

}
};
```

Java:

```
class Solution {
public int maximumPossibleSize(int[] nums) {

}
}
```

Python3:

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