Problem 3105: Longest Strictly Increasing or Strictly Decreasing Subarray

Problem Information

Difficulty: Easy
Acceptance Rate: 64.95%
Paid Only: No
Tags: Array

Problem Description

You are given an array of integers `nums`. Return _the length of the**longest** subarray of _`nums` _which is either**strictly increasing** or **strictly decreasing**_.

Example 1:

Input: nums = [1,4,3,3,2]

Output: 2

Explanation:

The strictly increasing subarrays of `nums` are `[1]`, `[2]`, `[3]`, `[3]`, `[4]`, and `[1,4]`.

The strictly decreasing subarrays of `nums` are `[1]`, `[2]`, `[3]`, `[3]`, `[4]`, `[3,2]`, and `[4,3]`.

Hence, we return `2`.

Example 2:

Input: nums = [3,3,3,3]

Output: 1

Explanation:

The strictly increasing subarrays of `nums` are `[3]`, `[3]`, `[3]`, and `[3]`.

The strictly decreasing subarrays of `nums` are `[3]`, `[3]`, `[3]`, and `[3]`.

Hence, we return `1`.

Example 3:

Input: nums = [3,2,1]

Output: 3

Explanation:

The strictly increasing subarrays of `nums` are `[3]`, `[2]`, and `[1]`.

The strictly decreasing subarrays of `nums` are `[3]`, `[2]`, `[1]`, `[3,2]`, `[2,1]`, and `[3,2,1]`.

Hence, we return `3`.

Constraints:

* `1 <= nums.length <= 50` * `1 <= nums[i] <= 50`

Code Snippets

C++:

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

}
};
```

Java:

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

```
}
}
```

Python3:

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