Problem 3350: Adjacent Increasing Subarrays Detection II

Problem Information

Difficulty: Medium
Acceptance Rate: 58.83%
Paid Only: No
Tags: Array, Binary Search

Problem Description

Given an array `nums` of `n` integers, your task is to find the **maximum** value of `k` for which there exist **two** adjacent subarrays of length `k` each, such that both subarrays are **strictly** **increasing**. Specifically, check if there are **two** subarrays of length `k` starting at indices `a` and `b` (`a < b`), where:

* Both subarrays `nums[a..a + k - 1]` and `nums[b..b + k - 1]` are **strictly increasing**. * The subarrays must be **adjacent** , meaning `b = a + k`.

Return the **maximum** _possible_ value of `k`.

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

Example 1:

Input: nums = [2,5,7,8,9,2,3,4,3,1]

Output: 3

Explanation:

* The subarray starting at index 2 is `[7, 8, 9]`, which is strictly increasing. * The subarray starting at index 5 is `[2, 3, 4]`, which is also strictly increasing. * These two subarrays are adjacent, and 3 is the **maximum** possible value of `k` for which two such adjacent strictly increasing subarrays exist.

Example 2:

Input: nums = [1,2,3,4,4,4,4,5,6,7]

Output: 2

Explanation:

* The subarray starting at index 0 is `[1, 2]`, which is strictly increasing. * The subarray starting at index 2 is `[3, 4]`, which is also strictly increasing. * These two subarrays are adjacent, and 2 is the **maximum** possible value of `k` for which two such adjacent strictly increasing subarrays exist.

Constraints:

* `2 <= nums.length <= 2 * 105` * `-109 <= nums[i] <= 109`

Code Snippets

C++:

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

}
};
```

Java:

```
class Solution {
public int maxIncreasingSubarrays(List<Integer> nums) {

}
}
```

Python3:

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