Problem 3349: Adjacent Increasing Subarrays Detection I

Problem Information

Difficulty: Easy
Acceptance Rate: 48.05%
Paid Only: No
Tags: Array

Problem Description

Given an array `nums` of `n` integers and an integer `k`, determine whether there exist **two** **adjacent** subarrays of length `k` such that both subarrays are **strictly** **increasing**. Specifically, check if there are **two** subarrays 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 `true` if it is _possible_ to find **two** such subarrays, and `false` otherwise.

Example 1:

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

Output: true

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, so the result is `true`.

Example 2:

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

Output: false

Constraints:

* `2 <= nums.length <= 100` * `1 < 2 * k <= nums.length` * `-1000 <= nums[i] <= 1000`

Code Snippets

C++:

```
class Solution {
public:
bool hasIncreasingSubarrays(vector<int>& nums, int k) {

}
};
```

Java:

```
class Solution {
public boolean hasIncreasingSubarrays(List<Integer> nums, int k) {

}
}
```

Python3:

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