Problem 2919: Minimum Increment Operations to Make Array Beautiful

Problem Information

Difficulty: Medium
Acceptance Rate: 34.47%
Paid Only: No
Tags: Array, Dynamic Programming

Problem Description

You are given a **0-indexed** integer array `nums` having length `n`, and an integer `k`.

You can perform the following **increment** operation **any** number of times (**including zero**):

* Choose an index `i` in the range `[0, n - 1]`, and increase `nums[i]` by `1`.

An array is considered **beautiful** if, for any **subarray** with a size of `3` or **more** , its **maximum** element is **greater than or equal** to `k`.

Return _an integer denoting the**minimum** number of increment operations needed to make _`nums` _**beautiful**._

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

Example 1:

Input: nums = [2,3,0,0,2], k = 4 **Output:** 3 **Explanation:** We can perform the following increment operations to make nums beautiful: Choose index i = 1 and increase nums[1] by 1 -> [2,4,0,0,2]. Choose index i = 4 and increase nums[4] by 1 -> [2,4,0,0,3]. Choose index i = 4 and increase nums[4] by 1 -> [2,4,0,0,4]. The subarrays with a size of 3 or more are: [2,4,0], [4,0,0], [0,0,4], [2,4,0,0], [4,0,0,4], [2,4,0,0,4]. In all the subarrays, the maximum element is equal to k = 4, so nums is now beautiful. It can be shown that nums cannot be made beautiful with fewer than 3 increment operations. Hence, the answer is 3.

Example 2:

Input: nums = [0,1,3,3], k = 5 **Output:** 2 **Explanation:** We can perform the following increment operations to make nums beautiful: Choose index i = 2 and increase nums[2] by 1 -> [0,1,4,3]. Choose index i = 2 and increase nums[2] by 1 -> [0,1,5,3]. The subarrays with a size of 3 or more are: [0,1,5], [1,5,3], [0,1,5,3]. In all the subarrays, the maximum element is equal to k = 5, so nums is now beautiful. It can be shown that nums cannot be made beautiful with fewer than 2 increment operations. Hence, the answer is 2.

Example 3:

Input: nums = [1,1,2], k = 1 **Output:** 0 **Explanation:** The only subarray with a size of 3 or more in this example is [1,1,2]. The maximum element, 2, is already greater than k = 1, so we don't need any increment operation. Hence, the answer is 0.

Constraints:

* `3 <= n == nums.length <= 105` * `0 <= nums[i] <= 109` * `0 <= k <= 109`

Code Snippets

C++:

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

}
};
```

Java:

```
class Solution {
public long minIncrementOperations(int[] nums, int k) {

}
}
```

Python3:

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