Problem 2111: Minimum Operations to Make the Array K-Increasing

Problem Information

Difficulty: Hard
Acceptance Rate: 39.94%
Paid Only: No
Tags: Array, Binary Search

Problem Description

You are given a **0-indexed** array `arr` consisting of `n` positive integers, and a positive integer `k`.

The array `arr` is called **K-increasing** if `arr[i-k] <= arr[i]` holds for every index `i`, where `k <= i <= n-1`.

* For example, `arr = [4, 1, 5, 2, 6, 2]` is K-increasing for `k = 2` because: * `arr[0] <= arr[2] (4 <= 5)` * `arr[1] <= arr[3] (1 <= 2)` * `arr[2] <= arr[4] (5 <= 6)` * `arr[3] <= arr[5] (2 <= 2)` * However, the same `arr` is not K-increasing for `k = 1` (because `arr[0] > arr[1]`) or `k = 3` (because `arr[0] > arr[3]`).

In one **operation** , you can choose an index `i` and **change** `arr[i]` into **any** positive integer.

Return _the**minimum number of operations** required to make the array K-increasing for the given _`k`.

Example 1:

Input: arr = [5,4,3,2,1], k = 1 **Output:** 4 **Explanation:** For k = 1, the resultant array has to be non-decreasing. Some of the K-increasing arrays that can be formed are [5,_**6**_ ,_**7**_ ,_**8**_ ,_**9**_], [_**1**_ ,_**1**_ ,_**1**_ ,_**1**_ ,1], [_**2**_ ,_**2**_ ,3,_**4**_ ,_**4**_]. All of them require 4 operations. It is suboptimal to change the array to, for example, [_**6**_ ,_**7**_ ,_**8**_ ,_**9**_ ,_**10**_] because it would take 5 operations. It can be shown that we cannot make the array K-increasing in less than 4 operations.

Example 2:

Input: arr = [4,1,5,2,6,2], k = 2 **Output:** 0 **Explanation:** This is the same example as the one in the problem description. Here, for every index i where 2 <= i <= 5, arr[i-2] <=**** arr[i]. Since the given array is already K-increasing, we do not need to perform any operations.

Example 3:

Input: arr = [4,1,5,2,6,2], k = 3 **Output:** 2 **Explanation:** Indices 3 and 5 are the only ones not satisfying arr[i-3] <= arr[i] for 3 <= i <= 5. One of the ways we can make the array K-increasing is by changing arr[3] to 4 and arr[5] to 5. The array will now be [4,1,5,_**4**_ ,6,_**5**_]. Note that there can be other ways to make the array K-increasing, but none of them require less than 2 operations.

Constraints:

* `1 <= arr.length <= 105` * `1 <= arr[i], k <= arr.length`

Code Snippets

C++:

```
class Solution {
public:
int kIncreasing(vector<int>& arr, int k) {

}
};
```

Java:

```
class Solution {
public int kIncreasing(int[] arr, int k) {

}
}
```

Python3:

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