Problem 2407: Longest Increasing Subsequence II

Problem Information

Difficulty: Hard
Acceptance Rate: 25.80%
Paid Only: No
Tags: Array, Divide and Conquer, Dynamic Programming, Binary Indexed Tree, Segment Tree, Queue, Monotonic Queue

Problem Description

You are given an integer array `nums` and an integer `k`.

Find the longest subsequence of `nums` that meets the following requirements:

* The subsequence is **strictly increasing** and * The difference between adjacent elements in the subsequence is **at most** `k`.

Return _the length of the**longest** **subsequence** that meets the requirements._

A **subsequence** is an array that can be derived from another array by deleting some or no elements without changing the order of the remaining elements.

Example 1:

Input: nums = [4,2,1,4,3,4,5,8,15], k = 3 **Output:** 5 **Explanation:** The longest subsequence that meets the requirements is [1,3,4,5,8]. The subsequence has a length of 5, so we return 5. Note that the subsequence [1,3,4,5,8,15] does not meet the requirements because 15 - 8 = 7 is larger than 3.

Example 2:

Input: nums = [7,4,5,1,8,12,4,7], k = 5 **Output:** 4 **Explanation:** The longest subsequence that meets the requirements is [4,5,8,12]. The subsequence has a length of 4, so we return 4.

Example 3:

Input: nums = [1,5], k = 1 **Output:** 1 **Explanation:** The longest subsequence that meets the requirements is [1]. The subsequence has a length of 1, so we return 1.

Constraints:

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

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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