Problem 3578: Count Partitions With Max-Min Difference at Most K

Problem Information

Difficulty: Medium
Acceptance Rate: 37.17%
Paid Only: No
Tags: Array, Dynamic Programming, Queue, Sliding Window, Prefix Sum, Monotonic Queue

Problem Description

You are given an integer array `nums` and an integer `k`. Your task is to partition `nums` into one or more **non-empty** contiguous segments such that in each segment, the difference between its **maximum** and **minimum** elements is **at most** `k`.

Return the total number of ways to partition `nums` under this condition.

Since the answer may be too large, return it **modulo** `109 + 7`.

Example 1:

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

Output: 6

Explanation:

There are 6 valid partitions where the difference between the maximum and minimum elements in each segment is at most `k = 4`:

* `[[9], [4], [1], [3], [7]]` * `[[9], [4], [1], [3, 7]]` * `[[9], [4], [1, 3], [7]]` * `[[9], [4, 1], [3], [7]]` * `[[9], [4, 1], [3, 7]]` * `[[9], [4, 1, 3], [7]]`

Example 2:

Input: nums = [3,3,4], k = 0

Output: 2

Explanation:

There are 2 valid partitions that satisfy the given conditions:

* `[[3], [3], [4]]` * `[[3, 3], [4]]`

Constraints:

* `2 <= nums.length <= 5 * 104` * `1 <= nums[i] <= 109` * `0 <= k <= 109`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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