Problem 1793: Maximum Score of a Good Subarray

Problem Information

Difficulty: Hard
Acceptance Rate: 64.32%
Paid Only: No
Tags: Array, Two Pointers, Binary Search, Stack, Monotonic Stack

Problem Description

You are given an array of integers `nums` **(0-indexed)** and an integer `k`.

The **score** of a subarray `(i, j)` is defined as `min(nums[i], nums[i+1], ..., nums[j]) * (j - i + 1)`. A **good** subarray is a subarray where `i <= k <= j`.

Return _the maximum possible**score** of a **good** subarray._

Example 1:

Input: nums = [1,4,3,7,4,5], k = 3 **Output:** 15 **Explanation:** The optimal subarray is (1, 5) with a score of min(4,3,7,4,5) * (5-1+1) = 3 * 5 = 15.

Example 2:

Input: nums = [5,5,4,5,4,1,1,1], k = 0 **Output:** 20 **Explanation:** The optimal subarray is (0, 4) with a score of min(5,5,4,5,4) * (4-0+1) = 4 * 5 = 20.

Constraints:

* `1 <= nums.length <= 105` * `1 <= nums[i] <= 2 * 104` * `0 <= k < nums.length`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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