Problem 689: Maximum Sum of 3 Non-Overlapping Subarrays

Problem Information

Difficulty: Hard
Acceptance Rate: 59.60%
Paid Only: No
Tags: Array, Dynamic Programming, Sliding Window, Prefix Sum

Problem Description

Given an integer array `nums` and an integer `k`, find three non-overlapping subarrays of length `k` with maximum sum and return them.

Return the result as a list of indices representing the starting position of each interval (**0-indexed**). If there are multiple answers, return the lexicographically smallest one.

Example 1:

Input: nums = [1,2,1,2,6,7,5,1], k = 2 **Output:** [0,3,5] **Explanation:** Subarrays [1, 2], [2, 6], [7, 5] correspond to the starting indices [0, 3, 5]. We could have also taken [2, 1], but an answer of [1, 3, 5] would be lexicographically larger.

Example 2:

Input: nums = [1,2,1,2,1,2,1,2,1], k = 2 **Output:** [0,2,4]

Constraints:

* `1 <= nums.length <= 2 * 104` * `1 <= nums[i] < 216` * `1 <= k <= floor(nums.length / 3)`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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