Problem 1696: Jump Game VI

Problem Information

Difficulty: Medium
Acceptance Rate: 46.23%
Paid Only: No
Tags: Array, Dynamic Programming, Queue, Heap (Priority Queue), Monotonic Queue

Problem Description

You are given a **0-indexed** integer array `nums` and an integer `k`.

You are initially standing at index `0`. In one move, you can jump at most `k` steps forward without going outside the boundaries of the array. That is, you can jump from index `i` to any index in the range `[i + 1, min(n - 1, i + k)]` **inclusive**.

You want to reach the last index of the array (index `n - 1`). Your **score** is the **sum** of all `nums[j]` for each index `j` you visited in the array.

Return _the**maximum score** you can get_.

Example 1:

Input: nums = [_1_ ,_-1_ ,-2,_4_ ,-7,_3_], k = 2 **Output:** 7 **Explanation:** You can choose your jumps forming the subsequence [1,-1,4,3] (underlined above). The sum is 7.

Example 2:

Input: nums = [_10_ ,-5,-2,_4_ ,0,_3_], k = 3 **Output:** 17 **Explanation:** You can choose your jumps forming the subsequence [10,4,3] (underlined above). The sum is 17.

Example 3:

Input: nums = [1,-5,-20,4,-1,3,-6,-3], k = 2 **Output:** 0

Constraints:

* `1 <= nums.length, k <= 105` * `-104 <= nums[i] <= 104`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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