



Explore

Problems

Interview

New

Contest

Discuss

Store

☆ Premium

Description

Solution

Discuss (612)

Submissions

i C++

1696. Jump Game VI

Medium

2073

76

Add to List

Share

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:

- $1 \leq \text{nums.length}, k \leq 10^5$
- $-10^4 \leq \text{nums}[i] \leq 10^4$

```
1 class Solution {
2 public:
3     int maxScore(vector<int> &nums, int k) {
4         int n = nums.size();
5         deque<int> q;
6         vector<int> dp(n, 0);
7         dp[0] = nums[0];
8         for (int i = 1; i < n; i++) {
9             while (!q.empty() && i - q.front() > k)
10                 q.pop_front();
11             dp[i] = dp[q.back()] + nums[i];
12             q.push_back(i);
13         }
14         return dp[n-1];
15     }
16 };
17
18
19
```

Testcase Run Code Re

Accepted Runtime

Your input

[1, 2]

Output

7

Expected

7

Accepted 62551

Submissions 141324

Problems

Pick One

< Prev

1696/2330

Next >

Example cases

?

Run C