

Problem 2297: Jump Game VIII

Problem Information

Difficulty: Medium

Acceptance Rate: 45.16%

Paid Only: Yes

Tags: Array, Dynamic Programming, Stack, Graph, Monotonic Stack, Shortest Path

Problem Description

You are given a **0-indexed** integer array `nums` of length `n`. You are initially standing at index `0`. You can jump from index `i` to index `j` where `i < j` if:

* `nums[i] <= nums[j]` and `nums[k] < nums[i]` for all indexes `k` in the range `i < k < j`, or *
* `nums[i] > nums[j]` and `nums[k] >= nums[i]` for all indexes `k` in the range `i < k < j`.

You are also given an integer array `costs` of length `n` where `costs[i]` denotes the cost of jumping **to** index `i`.

Return **the** **minimum** cost to jump to the index `n - 1`.

Example 1:

Input: `nums = [3,2,4,4,1]`, `costs = [3,7,6,4,2]` **Output:** 8 **Explanation:** You start at index 0. - Jump to index 2 with a cost of `costs[2] = 6`. - Jump to index 4 with a cost of `costs[4] = 2`. The total cost is 8. It can be proven that 8 is the minimum cost needed. Two other possible paths are from index 0 -> 1 -> 4 and index 0 -> 2 -> 3 -> 4. These have a total cost of 9 and 12, respectively.

Example 2:

Input: `nums = [0,1,2]`, `costs = [1,1,1]` **Output:** 2 **Explanation:** Start at index 0. - Jump to index 1 with a cost of `costs[1] = 1`. - Jump to index 2 with a cost of `costs[2] = 1`. The total cost is 2. Note that you cannot jump directly from index 0 to index 2 because `nums[0] <= nums[1]`.

****Constraints:****

* `n == nums.length == costs.length` * `1 <= n <= 105` * `0 <= nums[i], costs[i] <= 105`

Code Snippets

C++:

```
class Solution {
public:
    long long minCost(vector<int>& nums, vector<int>& costs) {

    }
};
```

Java:

```
class Solution {
    public long minCost(int[] nums, int[] costs) {

    }
}
```

Python3:

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