Problem 3693: Climbing Stairs II

Problem Information

Difficulty: Medium
Acceptance Rate: 63.60%
Paid Only: No
Tags: Array, Dynamic Programming

Problem Description

You are climbing a staircase with `n + 1` steps, numbered from 0 to `n`.

You are also given a **1-indexed** integer array `costs` of length `n`, where `costs[i]` is the cost of step `i`.

From step `i`, you can jump **only** to step `i + 1`, `i + 2`, or `i + 3`. The cost of jumping from step `i` to step `j` is defined as: `costs[j] + (j - i)2`

You start from step 0 with `cost = 0`.

Return the **minimum** total cost to reach step `n`.

Example 1:

Input: n = 4, costs = [1,2,3,4]

Output: 13

Explanation:

One optimal path is `0 -> 1 -> 2 -> 4`

Jump | Cost Calculation | Cost ---|---|--- 0 -> 1 | `costs[1] + (1 - 0)2 = 1 + 1` | 2 1 -> 2 | `costs[2] + (2 - 1)2 = 2 + 1` | 3 2 -> 4 | `costs[4] + (4 - 2)2 = 4 + 4` | 8 Thus, the minimum total cost is `2 + 3 + 8 = 13`

Example 2:

Input: n = 4, costs = [5,1,6,2]

Output: 11

Explanation:

One optimal path is `0 -> 2 -> 4`

Jump | Cost Calculation | Cost ---|---|--- 0 -> 2 | `costs[2] + (2 - 0)2 = 1 + 4` | 5 2 -> 4 | `costs[4] + (4 - 2)2 = 2 + 4` | 6 Thus, the minimum total cost is `5 + 6 = 11`

Example 3:

Input: n = 3, costs = [9,8,3]

Output: 12

Explanation:

The optimal path is `0 -> 3` with total cost = `costs[3] + (3 - 0)2 = 3 + 9 = 12`

Constraints:

* `1 <= n == costs.length <= 105■■■■■■■` * `1 <= costs[i] <= 104`

Code Snippets

C++:

```
class Solution {
public:
int climbStairs(int n, vector<int>& costs) {

}
};
```

Java:

```
class Solution {
public int climbStairs(int n, int[] costs) {

}
}
```

Python3:

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