**746. Min Cost Climbing Stairs: -**

Easy Accepted: 957.9K Submissions: 1.5M Acceptance Rate: 64.5%

You are given an integer array cost where cost[i] is the cost of ith step on a staircase. Once you pay the cost, you can either climb one or two steps.

You can either start from the step with index 0, or the step with index 1.

Return *the minimum cost to reach the top of the floor*.

**Example 1:**

**Input:** cost = [10,15,20]

**Output:** 15

**Explanation:** You will start at index 1.

- Pay 15 and climb two steps to reach the top.

The total cost is 15.

**Example 2:**

**Input:** cost = [1,100,1,1,1,100,1,1,100,1]

**Output:** 6

**Explanation:** You will start at index 0.

- Pay 1 and climb two steps to reach index 2.

- Pay 1 and climb two steps to reach index 4.

- Pay 1 and climb two steps to reach index 6.

- Pay 1 and climb one step to reach index 7.

- Pay 1 and climb two steps to reach index 9.

- Pay 1 and climb one step to reach the top.

The total cost is 6.

**Constraints:**

* 2 <= cost.length <= 1000
* 0 <= cost[i] <= 999

**Code: -**

class Solution {

public:

    int helper(vector<int> &cost, int ind, vector<int> &dp){

        int n = cost.size();

        // base case

        if(ind >= n)

            return 0;

        // dp found case

        if(dp[ind] != -1)

            return dp[ind];

        // recursive case

        int one = helper(cost, ind+1, dp);

        int two = helper(cost, ind+2, dp);

        // return from current state

        return dp[ind] = cost[ind] + min(one, two);

    }

    int minCostClimbingStairs(vector<int>& cost) {

        int n = cost.size();

        vector<int> dp(n, -1);

        return min(helper(cost, 0, dp), helper(cost, 1, dp));

    }

};

**T.C: - O(N)**

**S.C: - O(N)**