746. Min Cost Climbing Stairs

DescriptionHintsSubmissionsDiscussSolution

On a staircase, the `i`-th step has some non-negative cost `cost[i]` assigned (0 indexed).

Once you pay the cost, you can either climb one or two steps. You need to find minimum cost to reach the top of the floor, and you can either start from the step with index 0, or the step with index 1.

Example 1:

```
Input: cost = [10, 15, 20]
Output: 15
Explanation: Cheapest is start on cost[1], pay that cost and go to the top.
```

Example 2:

```
Input: cost = [1, 100, 1, 1, 1, 100, 1, 1, 100, 1]
Output: 6
Explanation: Cheapest is start on cost[0], and only step on 1s, skipping
cost[3].
```

Note:

1. `cost` will have a length in the range `[2, 1000]`.
2. Every `cost[i]` will be an integer in the range `[0, 999]`.

Seen this question in a real interview before?

- Difficulty:Easy
- Total Accepted:1.3K
- Total Submissions:2.8K
- Contributor:TopCoder111
-
- Subscribe to see which companies asked this question.

  Related Topics Dynamic Programming

  Similar Questions

  Climbing Stairs

```
class Solution {
public:
    int minCostClimbingStairs(vector<int>& cost) {
      int n = cost.size();
```

```
        if(n<=2) return min(cost[0],cost[1]);
        vector<int> optCost(n,INT_MAX);
        optCost[0] = cost[0];
        optCost[1] = cost[1];
        for(int i=2;i<n;++i)
        {
              int tmp = min(optCost[i-1],optCost[i-2]);
              optCost[i]=tmp+cost[i];
        }
        return min(optCost[n-1],optCost[n-2]);
}
};
```