## Sample Solution Discussion: Forum on Dynamic Programming

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].

```
def mincost(cost):
    n = len(cost)

    dp = [0 for _ in range(n + 1)]
    for i in range(2, n + 1):
        dp[i] = min(dp[i - 1] + cost[i - 1], dp[i - 2] + cost[i - 2])

    return dp[n]
```

Running time: O(n)