Stickler Thief :-

Stickler the thief wants to loot money from the houses arranged in a line. He cannot loot **two consecutive** houses and aims to **maximize** his total loot.
Given an array, **arr**[] where arr[i] represents the amount of money in the i-th house.
Determine the **maximum** amount he can loot.

Examples:

Input: arr[] = [6, 5, 5, 7, 4]
Output: 15
Explanation: Maximum amount he can get by looting 1st, 3rd and 5th house. Which is 6 + 5 + 4 = 15.

Input: arr[] = [1, 5, 3]
Output: 5
Explanation: Loot only 2nd house and get maximum amount of 5.

Input: arr[] = [4, 4, 4, 4]
Output: 8
Explanation: The optimal choice is to loot every alternate house. Looting the 1st and 3rd houses, or the 2nd and 4th, both give a maximum total of 4 + 4 = 8.

Constraints:
$1 \leq arr.size() \leq 10^5$
$1 \leq arr[i] \leq 10^4$

Try more examples

Company Tags

eg: I/p : arr[] = [6, 5, 5, 7, 4]
O/p : 15

dp[i][l]

l = 1 → loot
l = 0 → don't loot

dp[i][0] = max(dp[i+1][1], dp[i+1][0])

dp[i][1] = max(dp[i+1][1], arr[i] + dp[i+1][0])

dp[n-1][0] = 0

dp[n-1][1] = arr[n-1]

C++ (g++ 5.4) Start Timer

```cpp
// } Driver Code Ends

class Solution {
  public:
    int findMaxSum(vector<int>& arr) {
        // code here
        int n=arr.size();
        vector<vector<int>> dp(n+2, vector<int>(2, 0));
        dp[n][0]=0;
        dp[n][1]=arr[n-1];
        for(int i=n-1;i>=1;i--){
            dp[i][0]=max(dp[i+1][1], dp[i+1][0]);
            dp[i][1]=max(dp[i+1][1], arr[i-1]+dp[i+1][0]);
        }
        return max(dp[1][0], dp[1][1]);
    }
};

// } Driver Code Ends
```

T.C. O(n)

S.C. O(n)