**Minimum sum of subarray**

**Easy**Accuracy: 52.59% Submissions: 1501 Points: 2

Given an array of integers of size N, for all, i's [1, N], the task is to find the minimum subarray sum in the subarray [i, N].

**Input:**  
1. The first line of the input contains a single integerT denoting the number of test cases. The description of T test cases follows.  
2. The first line of each test case contains a single integer N.  
3. The second line contains N space-separated positive integers represents array.  
  
**Output:** For each test case, print N space-separated integers  
  
**Constraints:**  
1. 1 <= T <= 10  
2. 1 <= N <= 100000  
3. -10000 <= arr[i] <= 10000  
  
**Example:  
Input:**  
2  
3  
3 -1 -2  
5  
5 -3 -2 9 4

**Output:**  
-3 -3 -2  
-5 -5 -2 4 4

**Explanation:**  
Test case 1:

1. i = 1, subarray is [3, -1, -2], all possible subarrays are [3], [-1], [-2], [3, -1], [-1, -2], [3, -1, -2]. Minnimum sum is -3 [-1, -2].  
2. i = 2, subarray is [-1, -2], all possible subarrays are [-1], [-2], [-1, -2]. Minnimum sum is -3 [-1, -2].  
3. i = 3, subarray is [-2], all possible subarrays are [-2]. Minnimum sum is -2[-2].

#include <bits/stdc++.h>

using namespace std;

int main() {

    //code

    int t;

    cin>>t;

    while (t--) {

        int n;

        cin>>n;

        vector<int> v(n), ans(n);

        for (int i=0; i<n; i++) cin>>v[i];

        int dp[n];

        for (int i=0; i<n; i++) dp[i]=v[i];

        for (int i=0; i<n; i++) {

            for (int j=i+1; j<n; j++) {

                dp[j]=min(dp[j], dp[j-1]+v[j]);

            }

            int minm=INT\_MAX;

            for (int j=i; j<n; j++) minm=min(minm, dp[j]);

            ans[i]=minm;

            for (int j=i; j<n; j++) dp[j]=v[j];

        }

        for (auto x : ans) cout<<x<<" ";

        cout<<"\n";

    }

    return 0;

}