**Minimum Cost of Reducing Array**

**Difficulty: HARD**

**Avg. time to solve**

**15 min**

**Success Rate**

**80%**

**Problem Statement**

**Suggest Edit**

**You are given an array 'ARR' consisting of 'N' positive integers, and you need to reduce the size of the array to 1 by performing an operation several number of times. In a single operation, you can merge any two adjacent elements of the array, and the cost of merging will be equal to the sum of those two elements. Find the minimum cost of reducing the given array by performing this operation several number of times.**

**In a single merge operation, the two elements are removed, and their sum is inserted at its place, hence decreasing the size of the array by 1 after each operation. For eg: Consider the array A1, A2, Ai-2, Ai-1, Ai, Aj, Aj+1, Aj+2 ,,,, An. Let the operation be performed on two indices i and j, So after merging the array will look like A1, A2, Ai-2, Ai-1, Ai+Aj, Aj+1, Aj+2,,,, An.**

**Note:**

Note that the given operation will be performed only 'N'-1 times, where 'N' is the size of the given array.

**Input Format:**

The first line of the input contains an integer T denoting the number of test cases.

The first line of each test case contains the integer N, denoting the size of the sorted array.

The second line of each test case contains N space-separated integers denoting the array elements.

**Output Format:**

The only line of output of each test case should contain a single integer which denotes the minimum cost of merging the whole array.

**Constraints:**

1 <= T <= 50

1 <= N <= 100

1 <= ARR[i] <= 10^6

Time Limit: 1 sec

**Sample Input 1:**

1

3

1 3 7

**Sample Output 1:**

15

**Explanation For Sample Input 1:**

The optimal way of merging is as follows:

Merge (1, 3). The array becomes {4, 7}. Cost for this operation is 4.

Merge (4, 7). The array becomes {11}. Cost for this operation is 11.

Therefore the total cost of merging is 4 + 11 = 15.

**Sample Input 2:**

1

4

3 2 4 1

**Sample Output 2:**

20

**Explanation For Sample Input 2:**

The optimal way of merging is as follows:

1. Merge (3, 2), then array becomes {5, 4, 1}. Cost for this operation is 5.

2. Merge (4, 1), then array becomes {5, 5}. Cost for this operation is 5.

3. Merge (5, 5), then array becomes {10}. Cost for this operation is 10.

Therefore the total cost of merging is 5 + 5 + 10 = 20.

#include <bits/stdc++.h>

int dp[101][101];

int solve(int l, int r, vector<int> &arr) {

    if (l==r) return 0;

    if (r-l==1) return arr[r]+arr[l];

    if (dp[l][r]!=-1) return dp[l][r];

    int sum=0;

    for (int i=l; i<=r; i++) sum+=arr[i];

    int ans=INT\_MAX;

    for (int i=l; i<r; i++) {

        int left=solve(l, i, arr);

        int right=solve(i+1, r, arr);

        ans=min(ans, left+right+sum);

    }

    return dp[l][r]=ans;

}

int findMinCost(vector<int> &arr, int n){

    // Write your code here.

    memset(dp, -1, sizeof(dp));

    return solve(0, n-1, arr);

}