**Partition Equal Subset Sum: -**

Medium Accuracy: 30.24% Submissions: 198K+ Points: 4

Given an array **arr[]** of size **N**, check if it can be partitioned into two parts such that the sum of elements in both parts is the same.

**Example 1:**

**Input:** N = 4

arr = {1, 5, 11, 5}

**Output:** YES

**Explanation:**

The two parts are {1, 5, 5} and {11}.

**Example 2:**

**Input:** N = 3

arr = {1, 3, 5}

**Output:** NO

**Explanation:** This array can never be

partitioned into two such parts.

**Your Task:**  
You do not need to read input or print anything. Your task is to complete the function **equalPartition()** which takes the value N and the array as input parameters and returns 1 if the partition is possible. Otherwise, returns 0.

**Expected Time Complexity:** O(N\*sum of elements)  
**Expected Auxiliary Space:** O(N\*sum of elements)

**Constraints:**  
1 ≤ N ≤ 100  
1 ≤ arr[i] ≤ 1000  
N\*sum of elements ≤ 5\*106

**Code: -**

//{ Driver Code Starts

// Initial Template for C++

#include <bits/stdc++.h>

using namespace std;

// } Driver Code Ends

// User function Template for C++

class Solution{

public:

bool helper(int arr[], int ind, int target, vector<vector<int>> &vis){

//base case

if(ind == 0)

return target==0;

if(target == 0)

return 1;

//dp found case

if(vis[ind][target] != -1)

return vis[ind][target];

//recursive case

int nottake = 0, take = 0;

//don't consider that item

nottake = helper(arr, ind-1, target, vis);

if(nottake == 1)

return vis[ind][target] = nottake;

//consider that item

if(target >= arr[ind])

take = helper(arr, ind-1, target-arr[ind], vis);

//return from current state

return (vis[ind][target] = take + nottake);

}

int equalPartition(int N, int arr[]){

int sum = accumulate(arr, arr+N, 0);

if(sum % 2 != 0)

return 0;

int target = sum / 2;

vector<vector<int>> vis(N, vector<int>(target+1, -1));

return helper(arr, N-1, target, vis);

}

};

//{ Driver Code Starts.

int main(){

int t;

cin>>t;

while(t--){

int N;

cin>>N;

int arr[N];

for(int i = 0;i < N;i++)

cin>>arr[i];

Solution ob;

if(ob.equalPartition(N, arr))

cout<<"YES\n";

else

cout<<"NO\n";

}

return 0;

}

// } Driver Code Ends

**T.C: - O(N \* sum of items)**

**S.C: - O(N \* sum of items)**