

Talent Battle 100 Days Coding Series

Mahesh got a beautiful array named **A** as a birthday gift from his beautiful girlfriend Namratha. There are **N** positive integers in that array. Mahesh loved the array so much that he started to spend a lot of time on it everyday. One day, he wrote down all possible subsets of the array. Then for each subset, he calculated the sum of elements in that subset and wrote it down on a paper. Unfortunately, Mahesh lost the beautiful array :(. He still has the paper on which he wrote all subset sums. Your task is to rebuild beautiful array **A** and help the couple stay happy :)

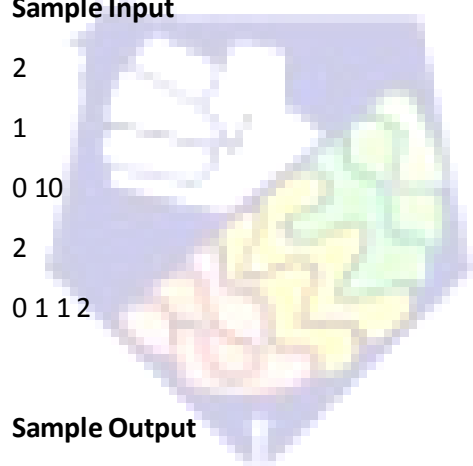
Input

The first line of the input contains an integer **T** denoting the number of test cases. First line of each test case contains one integer **N**, the number of elements in **A**. Second line of each test case contains 2^N integers, the values written on paper

Output

For each test case, output one line with **N** space separated integers in non-decreasing order.

Sample Input



```
2
1
0 10
2
0 1 1 2
```

Sample Output

```
10
1 1
```

C++

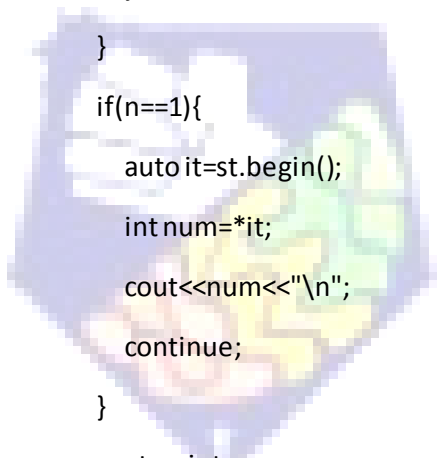
```
#include <bits/stdc++.h>

using namespace std;

int main(){
    int t;
    cin>>t;
```

Talent Battle 100 Days Coding Series

```
while(t--){
    int n;
    cin>>n;
    int end=pow(2,n);
    multiset<int>st;
    for(int i=0; i<end; i++){
        int x;
        cin>>x;
        if(x!=0){
            st.insert(x);
        }
    }
    if(n==1){
        auto it=st.begin();
        int num=*it;
        cout<<num<<"\n";
        continue;
    }
    vector<int>org;
    vector<int>sub_sums;
    for(int i=0; i<2; i++){
        auto it=st.begin();
        int x=*it;
        org.push_back(x);
        sub_sums.push_back(x);
        st.erase(it);
    }
    sub_sums.push_back(org[0]+org[1]);
    st.erase(org[0]+org[1]);
}
```



TalentBattle

Talent Battle 100 Days Coding Series

```
for(int i=2; i<n; i++){
    auto it=st.begin();
    int number=*it;
    st.erase(it);
    org.push_back(number);
    vector<int> newsums;
    for(int sum:sub_sums){
        st.erase(number+sum);
        newsums.push_back(number+sum);
    }
    for(int x:newsums){
        sub_sums.push_back(x);
    }
    sub_sums.push_back(number);
}
for(int x:org){
    cout<<x<<" ";
}
cout<<"\n";
}
return 0;
}
```

Java

```
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.Map;
import java.util.Scanner;
```

Talent Battle 100 Days Coding Series

```
import java.util.TreeMap;

public class Main {

    public static void main(String[] args) throws IOException {

        Scanner in = new Scanner(System.in);

        PrintWriter out = new PrintWriter(System.out);

        int t = in.nextInt();
        while(t-->0) {
            int n = in.nextInt();
            Map<Integer, Integer> subsets = new TreeMap<Integer, Integer>();
            for(int i = 0; i < 1 << n; i++) {
                int s = in.nextInt();
                Integer prev = subsets.get(s);
                if (prev == null)
                    prev = 0;
                subsets.put(s, prev + 1);
            }

            subsets.remove(0);

            ArrayList<Integer> nums = new ArrayList<Integer>();
            for(int i = 0; i < n; i++) {
                nums.add(subsets.keySet().iterator().next());
                for(int mask = 1 << nums.size() - 1; mask < 1 << nums.size(); mask++) {
                    int s = 0;
                    for(int j = 0; j <= i; j++)
```

Talent Battle 100 Days Coding Series

```
        if ((mask & (1 << j)) > 0)
            s += nums.get(j);
        Integer cnt = subsets.get(s);
        if (cnt == 1)
            subsets.remove(s);
        else
            subsets.put(s, cnt - 1);
    }
}
for(int i = 0; i < n; i++)
    out.print(nums.get(i) + " ");
out.println();
}
in.close();
out.close();
}
}
```

Python

```
def binarySearch(arr, l, r, x):
    if r >= l:
        mid = l + (r - l) // 2
        if arr[mid] == x:
            return mid
        elif arr[mid] > x:
            return binarySearch(arr, mid + 1, r, x)
```

Talent Battle 100 Days Coding Series

else:

 return binarySearch(arr, l, mid-1, x)

else:

 return -1

for _ in range(int(input().strip())):

 n = int(input().strip())

 sub_sums_list = list(map(int, input().strip().split()))

 sub_sums = sorted(sub_sums_list, reverse=True)

 sub_sums.pop()

 original_set = []

 to_be_removed = []

 while len(original_set) < n:

 element = sub_sums.pop()

 original_set.append(element)

 will_be_removed = [element]

 for rem_val in to_be_removed:

 new_rem_val = rem_val + element

 will_be_removed.append(new_rem_val)

 idx = binarySearch(sub_sums, 0, len(sub_sums) - 1, new_rem_val)

 sub_sums.pop(idx)

 to_be_removed += will_be_removed

print(*sorted(original_set), sep=" ")