

REC-CIS

# GE23131-Programming Using C-2024

Quiz navigation



Show one page at a time

Finish review

Status	Finished
Started	Monday, 23 December 2024, 5:33 PM
Completed	Saturday, 21 December 2024, 2:06 PM
Duration	2 days 3 hours

Question 1

Correct

Marked out of 5.00

Flag question

Sunny and Johnny like to pool their money and go to the ice cream parlor. Johnny never buys the same flavor that Sunny does. The only other rule they have is that they spend all of their money.

Given a list of prices for the flavors of ice cream, select the two that will cost all of the money they have.

For example, they have  $m = 6$  to spend and there are flavors costing **cost** = [1, 2, 3, 4, 5, 6]. The two flavors costing 1 and 5 meet the criteria. Using 1-based indexing, they are at indices 1 and 4.

Function Description

## REC-CIS

Complete the code in the editor below. It should return an array containing the indices of the prices of the two flavors they buy.

It has the following:

- `m`: an integer denoting the amount of money they have to spend
- `cost`: an integer array denoting the cost of each flavor of ice cream

**Input Format**

The first line contains an integer, `t`, denoting the number of trips to the ice cream parlor. The next `t` sets of lines each describe a visit. Each trip is described as follows:

1. The integer `m`, the amount of money they have pooled.
2. The integer `n`, the number of flavors offered at the time.
3. `n` space-separated integers denoting the cost of each flavor: `cost[cost[1], cost[2], . . . , cost[n]]`.

**Note:** The index within the cost array represents the flavor of the ice cream purchased.

**Constraints**

## REC-CIS

- $1 \leq t \leq 50$
- $2 \leq m \leq 10^4$
- $2 \leq n \leq 10^4$
- $1 \leq \text{cost}[i] \leq 10^4, " i \in [1, n]$
- There will always be a unique solution.

**Output Format**

For each test case, print two space-separated integers denoting the indices of the two flavors purchased, in ascending order.

**Sample Input**

```
2
4
5
1 4 5 3 2
4
4
```

REC-CIS

4

2 2 4 3

**Sample Output**

1 4

1 2

**Explanation**

Sunny and Johnny make the following two trips to the parlor:

1. The first time, they pool together  $m = 4$  dollars. Of the five flavors available that day, flavors **1** and **4** have a total cost of  $1 + 3 = 4$ .
2. The second time, they pool together  $m = 4$  dollars. TOf the four flavors available that day, flavors **1** and **2** have a total cost of  $2 + 2 = 4$ .

**Answer:** (penalty regime: 0 %)

1 #include&lt;stdio.h&gt;

## REC-CIS

```
1 #include<stdio.h>
2 int main()
3 {
4     int t,m,n,c=0;
5     scanf("%d",&t);
6     for(int i=0;i<t;i++){
7         c=0;
8         scanf("%d\n%d",&m,&n);
9         int arr[n];
10        for(int j=0;j<n;j++){
11            scanf("%d",&arr[j]);
12        }
13        for(int a=0;a<n-1;a++){
14            for(int b=a+1;b<n;b++){
15                if(arr[a]+arr[b]==m){
16                    printf("%d %d\n",a+1,b+1);
17                    c=1;break;
18                }
19            }if(c==1)break;
20        }
21    }
22    return 0;
23 }
```

	Input	Expected	Got	
✓	2	1 4	1 4	✓

## REC-CIS

✓	2	1 4	1 4	✓
	4	1 2	1 2	
	5			
	1 4 5 3 2			
	4			
	4			
	2 2 4 3			

Passed all tests! ✓

Question **2**

Correct

Marked out of 5.00

🚩 Flag question

Numeros the Artist had two lists that were permutations of one another. He was very proud. Unfortunately, while transporting them from one exhibition to another, some numbers were lost out of the first list. Can you find the missing numbers?

As an example, the array with some numbers missing, **arr** = [7, 2, 5, 3, 5, 3]. The original array of numbers **brr** = [7, 2, 5, 4, 6, 3, 5, 3]. The numbers missing are [4, 6].

### Notes

- If a number occurs multiple times in the lists, you must ensure that the frequency of that number in both lists is the same. If that is not the case, then it is also a missing number.

## REC-CIS

- If a number occurs multiple times in the lists, you must ensure that the frequency of that number in both lists is the same. If that is not the case, then it is also a missing number.
- You have to print all the missing numbers in ascending order.
- Print each missing number once, even if it is missing multiple times.
- The difference between maximum and minimum number in the second list is less than or equal to **100**.

Complete the code in the editor below. It should return an array of missing numbers.

It has the following:

- `arr`: the array with missing numbers
- `brr`: the original array of numbers

### Input Format

There will be four lines of input:

***n*** - the size of the first list, ***arr***

The next line contains ***n*** space-separated integers ***arr[i]***

***m*** - the size of the second list, ***brr***

The next line contains ***m*** space-separated integers ***brr[i]***

REC-CIS

**Constraints**

- $1 \leq n, m \leq 2 \times 10^5$
- $n \leq m$
- $1 \leq brr[i] \leq 2 \times 10^4$
- $X_{max} - X_{min} < 101$

**Output Format**

Output the missing numbers in ascending order.

**Sample Input**

10

203 204 205 206 207 208 203 204 205 206

13

203 204 204 205 206 207 205 208 203 206 205 206 204



REC-CIS

**Sample Output**

204 205 206

**Explanation**

**204** is present in both arrays. Its frequency in **arr** is **2**, while its frequency in **brr** is **3**. Similarly, **205** and **206** occur twice in **arr**, but three times in **brr**. The rest of the numbers have the same frequencies in both lists.

**Answer:** (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main()
3 {
4     int n,m,c,c1=0,co;
5     scanf("%d",&n);
6     int arr[n];
7     for(int a=0;a<n;a++){
8         scanf("%d",&arr[a]);
9     }
10    scanf("%d",&m);
11    int brr[m],ans[m];
12    for(int b=0;b<m;b++){
13        scanf("%d",&brr[b]);
14    }
```

## REC-CIS

```
14 }
15 for(int j=0;j<m;j++)
16 {
17     c=0;
18     for(int i=0;i<n;i++){
19         if(arr[i]==brr[j]){
20             c=1;
21             arr[i]=-1;
22             break;
23         }
24     }
25     if(c==0){
26         ans[c1]=brr[j];
27         c1++;
28     }
29 }
30 for(int a=0;a<c1;a++){
31     co=0;
32     for(int b=0;b<c1;b++){
33         if(ans[b]<ans[a])
34             co++;
35     }
36     int temp=ans[a];
37     ans[a]=ans[co];
38     ans[co]=temp;
39 }
40 for(int i=0;i<c1;i++)
41     printf("%d ",ans[i]);
42 return 0;
43 }
```





## REC-CIS

**Explanation 1**

In the first test case,  **$arr[2] = 4$**  is between two subarrays summing to **2**.

In the second case,  **$arr[0] = 2$**  is between two subarrays summing to **0**.

In the third case,  **$arr[2] = 2$**  is between two subarrays summing to **0**.

**Answer:** (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main()
3 {
4     int t,n,Is,rs,m;
5     scanf("%d",&t);
6     for(int i=0;i<t;i++){
7         Is=0;
8         rs=0;
9         scanf("%d",&n);
10        int arr[n];
11        for(int j=0;j<n;j++){
12            scanf("%d",&arr[j]);
13        }
14    }
```

REC-CIS

**Answer:** (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main(){
3     int t;
4     scanf("%d",&t);
5     while (t--){
6         int n;
7         scanf("%d",&n);
8         int arr[n],ts=0,ls=0;
9         for(int i=0;i<n;i++){
10             scanf("%d",&arr[i]);
11             ts+=arr[i];
12         }
13         int found =0;
14         for(int i=0;i<n;i++){
15             ts -=arr[i];
16             if(ls==ts){
17                 found =1;
18                 break;
19             }
20             ls += arr[i];
21         }
22     }
23     if(found){
24         printf("YES\n");
25     }else{
26         printf("NO\n");
27     }
28 }
```



4

