GE23131-Programming Using C-2024

Quiz navigation



Show one page at a time

Finish review

Started Monday, 23 December 2024, 5:33 PM

Completed Friday, 20 December 2024, 7:28 PM

Duration 2 days 22 hours

Question 1

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

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: on integer denoting the amount of money than be a leading to

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:

- 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

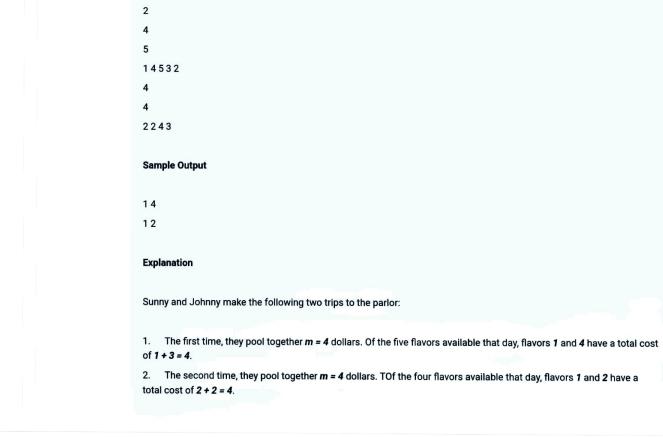
- . 1 ≤ t ≤ 50
- · 2 ≤ m ≤ 10⁴
- 2 ≤ n ≤ 10⁴
- 1 ≤ cost[i] ≤ 10⁴, " i Î [1, n]
- There will always be a unique solution.

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

Sample Output

Explanation

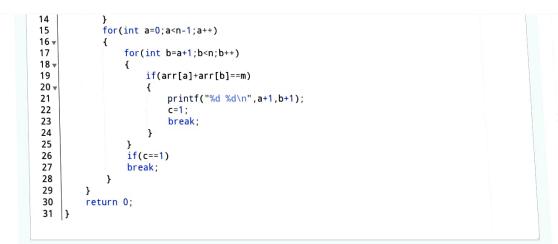
Curry and Johnny make the following two trine to the parlor:



Sample Output Explanation Sunny and Johnny make the following two trips to the parlor:

```
REC-CIS
```

Answer: (penalty regime: 0 %) #include<stdio.h> int main() 2 3 ▼ { int t,m,n,c=0; scanf("%d",&t); for(int i=0;i<t;i++) 7 ▼ 8 c=0; scanf("%d\n%d",&m,&n); 10 int arr[n]; 11 for(int j=0; j<n; j++)</pre> 12 ▼ 13 scanf("%d",&arr[j]); 14 15 for(int a=0;a<n-1;a++) 16 ▼ 17 for(int b=a+1;b<n;b++)</pre> 18 ▼ 19 if(arr[a]+arr[b]==m) 20 ▼ 21 printf("%d %d\n",a+1,b+1); 22 c=1; 23 break; 24 25 if(c==1) 26 27 break; 28 29 30 return 0; 31 }





Passed all tests! 🗸

REC-CIS

Correct Marked out of 5.00 Flag question

Question 2

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].

If a number occurs multiple times in the lists, you must ensure that the frequency of that number in both lists is the

Notes

- 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.
- Tou have to print all the missing numbers in assertaing 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

Sin the original array of hambers

C-CIS	
	There will be four lines of input:
	n - the size of the first list, arr
	The next line contains <i>n</i> space-separated integers <i>arr[i]</i>
	m - the size of the second list, brr
	The next line contains m space-separated integers $brr[i]$
	Constraints
	· 1≤n, m≤2x10 ⁵
	· n≤m
	1 ≤ brr[i] ≤ 2 x 10 ⁴
	$X_{max} - X_{min} < 101$
	Output Format
	Output the missing numbers in ascending order.
	Sample Input
	10

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 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.

Output the missing numbers in ascending order.

```
Answer: (penalty regime: 0 %)
      #include<stdio.h>
       int main()
           int n,m,c,c1=0,co;
           scanf("%d",&n);
           int arr[n];
           for(int a=0;a<n;a++)</pre>
   8 ₹
               scanf("%d",&arr[a]);
  10
  11
           scanf("%d",&m);
           int brr[m],ans[m];
  12
  13
           for(int b=0;b<m;b++)</pre>
  14 ▼
  15
               scanf("%d",&brr[b]);
  16
  17
           for(int j=0;j<m;j++)</pre>
  18 ₩
  19
               c=0:
  20
               for(int i=0;i<n;i++)
  21 ▼
  22
                    if(arr[i]==brr[j])
  23 ▼
  24
                        c=1;
  25
                        arr[i]=-1;
  26
                        break;
  27
  28
  29
               if(c==0)
  30 ▼
  31
                    ans[c1]=brr[j];
  32
                    c1++;
  33
  34
  35
           for(int a=0;a<c1;a++)
```

```
19
            c=0;
20
            for(int i=0;i<n;i++)</pre>
21 ▼
                 if(arr[i]==brr[j])
22
23 ▼
24
                     c=1;
25
                     arr[i]=-1;
26
                     break;
27
28
29
            if(c==0)
30 ₹
31
                 ans[c1]=brr[j];
32
                 c1++;
33
34
35
        for(int a=0;a<c1;a++)</pre>
36 ▼
37
             co=0:
38
             for(int b=0;b<c1;b++)
39 ₹
40
                 if(ans[b]<ans[a])</pre>
41
                 co++;
42
43
             int temp=ans[a];
44
             ans[a]=ans[co];
45
             ans[co]=temp;
46
47
         for(int i=0;i<c1;i++)
         printf("%d ",ans[i]);
48
49
         return 0;
50 }
```

```
31
32
                ans[c1]=brr[j];
                c1++;
33
34
35
        for(int a=0;a<c1;a++)
36 ₹
37
            co=0;
38
            for(int b=0;b<c1;b++)</pre>
39 ▼
40
                if(ans[b]<ans[a])</pre>
41
                co++;
42
43
            int temp=ans[a];
44
            ans[a]=ans[co];
45
            ans[co]=temp;
46
47
        for(int i=0;i<c1;i++)
48
        printf("%d ",ans[i]);
49
        return 0;
50 }
```

	Input	Expected	Got	
~	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	204 205 206	204 205 206	~

Correct

Marked out of
5.00

Frag question

Ouestion 3

Watson gives Sherlock an array of integers. His challenge is to find an element of the array such that the sum of all elements to the left is equal to the sum of all elements to the right. For instance, given the array arr = [5, 6, 8, 11], 8 is between two subarrays that sum to 11. If your starting array is [1], that element satisfies the rule as left and right sum to 0.

You will be given arrays of integers and must determine whether there is an element that meets the criterion.

Complete the code in the editor below. It should return a string, either YES if there is an element meeting the criterion or

It has the following:

arr: an array of integers

Input Format

NO otherwise.

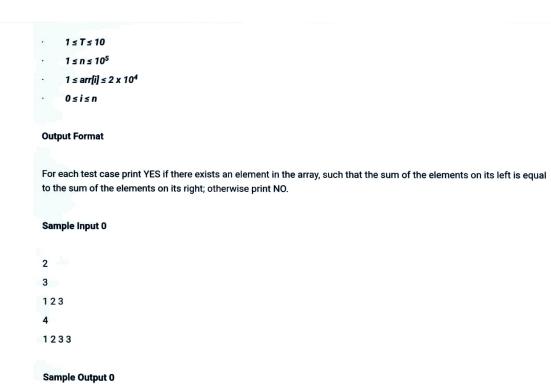
The first line contains $\emph{\textbf{T}}$, the number of test cases.

The next ${\it T}$ pairs of lines each represent a test case.

- The first line contains **n**, the number of elements in the array **arr**.
- The second line contains n space-separated integers arr[i] where $0 \le i < n$.

Constraints

1 ≤ T ≤ 10



NO YES

KE	Ç.

- YES

YES YES

Explanation 1

Sample Input 1

3 5 11411

2000

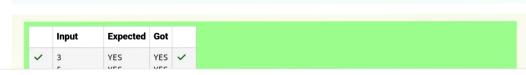
0020

Sample Output 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.

REC-CIS

```
Answer: (penalty regime: 0 %)
      #include<stdio.h>
       int main()
   3 ₹ {
           int t,n,Is,rs,m;
           scanf("%d",&t);
           for(int i=0;i<t;i++)
   8
               Is=0:
               rs=0;
  10
               scanf("%d",&n);
  11
               int arr[n];
               for(int j=0; j<n; j++)</pre>
  12
               scanf("%d",&arr[j]);
  13
  14
               m=n/2:
  15
               if(arr[m]==0)
  16 ▼
  17
                   for(m=0;arr[m]==0 && m<n;m++);
  18
  19
               for(int j=0; j<=m; j++)</pre>
  20
               Is=Is+arr[j];
  21
               for(int j=m; j<n; j++)</pre>
  22
               rs=rs+arr[j];
  23
               printf("%s\n",(Is==rs)?"YES":"NO");
  24
  25
           return 0;
  26 }
```



```
17
                 for(m=0;arr[m]==0 && m<n;m++);
18
19
             for(int j=0;j<=m;j++)</pre>
20
             Is=Is+arr[j];
21
             for(int j=m;j<n;j++)</pre>
22
             rs=rs+arr[j];
             printf("%s\n",(Is==rs)?"YES":"NO");
23
24
25
        return 0;
26 }
```

