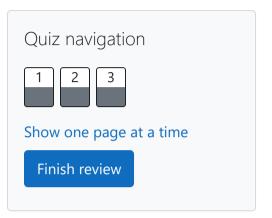
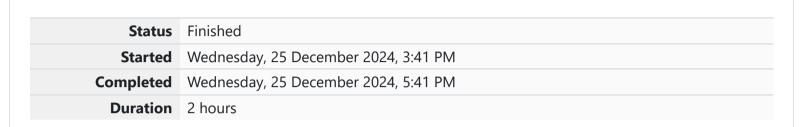
GE23131-Programming Using C-2024





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 1 meet the criteria. Using 1-based indexing, they are at indices 1 and 1.

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:

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

- $1 \le t \le 50$
- $2 \le m \le 10^4$
- $2 \le n \le 10^4$
- $1 \le \cos[i] \le 10^4$, " $i \hat{I} [1, n]$
- · 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 Output

14

12

Explanation

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

- 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.
- 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<stdio.h>

```
scanf("%d",&a);
 6 1
         for(int i=0;i<a;i++){</pre>
 7
             c=0;
             scanf("%d\n%d",&b,&d);
 8
 9
             int arr[d];
             for(int j=0;j<d;j++){</pre>
10 🔻
                  scanf("%d",&arr[j]);
11
12
             for(int x=0;x<d-1;x++){</pre>
13 🔻
                  for(int y=x+1;y<d;y++){</pre>
14 🔻
                      if(arr[x]+arr[y]==b){
15 🔻
                           printf("%d %d\n",x+1,y+1);
16
17
                           c=1;
18
                           break;
19
20
                  if(c==1)
21
22
                  break;
23
24
25
26
```

	Input	Expected	Got	
~	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! <

Marked out of 5.00

Flag question

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.
- 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]*

Constraints

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

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

Sample Output

204 205 206

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 %)

```
#include<stdio.h>
 2 v int main(){
 3
         int n,m,c,c1=0,co;
         scanf("%d",&n);
 4
         int arr[n];
         for(int a=0;a<n;a++){</pre>
 6
              scanf("%d",&arr[a]);
 8
         scanf("%d",&m);
 9
         int brr[m],ans[m];
10
         for(int b=0;b<m;b++){</pre>
11 1
              scanf("%d",&brr[b]);
12
13
         for(int j=0;j<m;j++)</pre>
14
15 1
              c=<mark>0</mark>;
16
              for(int i=0;i<n;i++){</pre>
17 1
                   if(arr[i]==brr[j])
18
19
20
                        c=1;
21
                        arr[i]=-1;
22
                        break;
23
24
              if(c==0){
25 1
                   ans[c1]=brr[j];
26
27
                   c1++;
28
29
         for(int a=0;a<c1;a++)</pre>
30
31 1
32
              co=<mark>0;</mark>
33 ▼
              for(int b=0:b<c1:b++){
```

	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	205 206 204	×

Your code must pass all tests to earn any marks. Try again.

Show differences

Question **3**

Incorrect

Marked out of 5.00

▼ Flag question

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], array 8 is between two subarrays that sum to array 11. If your starting array is array 11, that element satisfies the rule as left and right sum to array 11.

Complete the code in the editor below. It should return a string, either YES if there is an element meeting the criterion or NO otherwise.

It has the following:

· arr: an array of integers

Input Format

The first line contains **T**, the number of test cases.

The next **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
- $1 \le n \le 10^5$
- $1 \leq arr[i] \leq 2 \times 10^4$
- $0 \le i \le n$

Output Format

For each test case print YES if there exists an element in the array, such that the sum of the elements on its left is equal to the sum of the elements on its right; otherwise print NO.

2 3 123 4 1233 Sample Output 0 NO YES **Explanation 0** For the first test case, no such index exists. For the second test case, **arr[0]** + **arr[1]** = **arr[3]**, therefore index **2** satisfies the given conditions. Sample Input 1 3 5 11411

0020

Sample Output 1

YES

YES

YES

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 v int main(){
        int t,n,is,rs,n;
 3
        scanf("%d",&t);
        for( int i = 0 ; i < t ; i++ ){</pre>
 5 ▼
            rs = 0;
            is = 0;
            scanf("%d",&n);
            int arr[n];
            for ( int j = 0; j < n ; j++ ){
10 🔻
                scanf("%d",&arr[j]);
11
12
13
            m = n / 2;
14
15
```

```
Syntax Error(s)
__tester__.c: In function 'main':
__tester__.c:3:19: error: redeclaration of 'n' with no linkage
   3 | int t,n,is,rs,n;
tester .c:3:11: note: previous declaration of 'n' with type 'int'
   3 | int t,n,is,rs,n;
           ^
__tester__.c:13:9: error: 'm' undeclared (first use in this function)
  13
             m = n / 2;
tester .c:13:9: note: each undeclared identifier is reported only once for
each function it appears in
__tester__.c:3:16: error: variable 'rs' set but not used [-Werror=unused-but-
set-variable]
    3 |
        int t,n,is,rs,n;
__tester__.c:3:13: error: variable 'is' set but not used [-Werror=unused-but-
set-variable]
    3 |
         int t,n,is,rs,n;
cc1: all warnings being treated as errors
```

Finish review