NIVETHA W 240801233 ECE D

Question Text 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: 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.

Question text 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]. 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]*

m - the size of the second list, brr

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

Constraints

- $1 \le n, m \le 2 \times 10 5$
- · n ≤ m
- $1 \le brr[i] \le 2 \times 10 4$
- · Xmax Xmin < 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

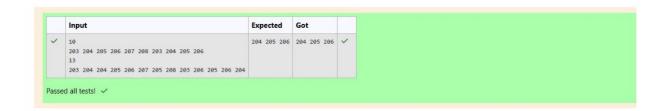
204 205 206

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.

```
1 |#include<stdio.h>
      int main(){
           int n,m,c1=0,co,c;
           scanf("%d",&n);
int arr[n];
for(int a=0;a<n;a++){</pre>
                scanf("%d",&arr[a]);
          int brr[m];
          int ans[m];
for(int b=0;b<m;b++){
                 scanf("%d",&brr[b]);
15
16
17
           for(int j=0;j<m;j++)
                 for(int i=0;i<n;i++){
    if(arr[i]==brr[j]){</pre>
21
                           break:
25
26
27
           if(c==0)
           ans[c1]=brr[j];
30
31
32
           for(int a=0;a<c1;a++)
                for(int b=0;b<c1;b++){
35
                     if(ans[b]<ans[a])
36
37
                int temp=ans[a];
ans[a]=ans[co];
ans[co]=temp;
39
40
41
42
           for(int i=0;i<c1;i++)
43
           printf("%d ",ans[i]);
            return 0;
```



Question text 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.

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 1 733 5 1 1 4 1 1 4 2 0 0 0 4 0 0 2 0 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.

	Input	Expected	Got	
~		YES	YES	~
		YES	YES	
	11411	YES	YES	
	2000			
	0020			
	2	NO	NO	~
		YES	YES	
	1 2 3			
	1233			
	1177			