

Programming Using C

week07 practice session coding

Name:K.Kamaleshwaran

Department:AIML-'B'

Roll No.:242501079

Harry and Johnny like to spend their money and go to the ice cream parlor. Johnny never buys the same flavor that Harry does. The only other rule they have is that they spend all of their money.
There is a list of prices for the flavors of ice cream, where the first value is the price of the first flavor and the last value is the price of the last flavor.
For example, they have an array of prices and there are 5 flavors. The first flavor costs 1 and the last flavor costs 5. The two flavors costing 1 and 5 are the only ones that they can afford to buy.

Function Description

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

It has the following parameters:
1. An integer array of prices of the flavors.
2. An integer array of prices of the flavors.

Input Format

The first line contains an integer, n , denoting the number of flavors of ice cream. The second line contains n space-separated integers denoting the prices of the flavors.

- 1. The integer n is the number of flavors of ice cream.
- 2. The integer m is the number of flavors of ice cream.
- 3. An array of n space-separated integers denoting the prices of the flavors: p_0, p_1, \dots, p_{n-1} .

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

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq m \leq 10^5$
- $1 \leq p_i \leq 10^5$
- $1 \leq p_i \leq 10^5$

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

5
1 2 3 4 5
1 2 3 4 5

Sample Output

1 4
2 3

Explanation

Harry and Johnny order the following two flavors: the first.

- 1. The first two flavors together cost 4 dollars. (The first flavor costs 1 dollar, the second flavor costs 3 dollars.)
- 2. The second two flavors together cost 4 dollars. (The first flavor costs 2 dollars, the second flavor costs 2 dollars.)

Answer: (1, 4) (2, 3)

```
1 def solve():  
2     n = int(input())  
3     p = list(map(int, input().split()))  
4     m = int(input())  
5     for i in range(n):  
6         for j in range(i+1, n):  
7             if p[i] + p[j] == m:  
8                 print(i, j)  
9                 return  
10    print(-1, -1)
```

Input	Expected Output	Obs
5	1 4	1 4
5	2 3	2 3
5	1 4	1 4
5	2 3	2 3

Percent of correct: 100%

As an example, the array with unique numbers missing `arr = [7, 2, 3, 4, 5, 6, 8]`. The original array of numbers `arr = [7, 2, 3, 4, 5, 6, 8]`. The numbers missing are `[1, 9]`.

Notes

If a number is an multiple then in the list, you must ensure that the frequency of that number 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 array is `100` so the size of the array is `101`.

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

It has the following:

`arr` the array with missing numbers

`len` the original array of numbers

Input Format

There will be four lines of input:

• The size of the list `len`

• The array `arr` consisting of space-separated integers `arr[]`

• The size of the second list `len`

• The array `arr` consisting of space-separated integers `arr[]`

Constraints

$1 \leq n \leq 2 \times 10^5$

$n \leq m$

$1 \leq arr[i] \leq 2 \times 10^5$

$abs(arr[i] - arr[j]) \geq 1$

Output Format

Output the missing numbers in ascending order.

Sample Input

```
10
101 102 103 104 105 106 107 108 109 110 111
10
107 108 109 110 111 112 113 114 115 116 117 118 119 120 121
```

Sample Output

```
101 111 116
```

Explanation

`101` is present in both arrays. Its frequency in `arr1` is `2` while its frequency in `arr2` is `1`. Similarly, `109` and `116` are not there in `arr2`. Therefore, these numbers have the same frequency in both lists.

Answer (Specialty program 0.7%)

```
1 // C++ implementation of the above approach
2 #include <iostream>
3 using namespace std;
4 // Function to find the missing numbers
5 void findMissingNumbers(int arr1[], int n1, int arr2[], int n2)
6 {
7     // Create a map to store the frequency of each element in arr1
8     map<int, int> mp1;
9     // Traverse arr1 and store the frequency of each element in mp1
10    for (int i = 0; i < n1; i++)
11        mp1[arr1[i]]++;
12    // Create a map to store the frequency of each element in arr2
13    map<int, int> mp2;
14    // Traverse arr2 and store the frequency of each element in mp2
15    for (int i = 0; i < n2; i++)
16        mp2[arr2[i]]++;
17    // Traverse mp1 and mp2 to find the missing numbers
18    vector<int> ans;
19    for (int i = 0; i < n1; i++)
20    {
21        // If the frequency of arr1[i] is greater than the frequency of arr2[i], then it is a missing number
22        if (mp1[arr1[i]] > mp2[arr1[i]])
23            ans.push_back(arr1[i]);
24    }
25    // Sort the missing numbers in ascending order
26    sort(ans.begin(), ans.end());
27    // Print the missing numbers
28    for (int i = 0; i < ans.size(); i++)
29        cout << ans[i] << " ";
30    cout << endl;
31 }
```

Input	Expected	Got
10 101 102 103 104 105 106 107 108 109 110 111 10 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121	101 111 116	101 111 116

Given an array of integers. The challenge is to find a subset of the array such that the sum of all elements in the left is equal to the sum of all elements in the right. For example, given the array $arr = [1, 2, 3, 4, 5]$, if a subset has values that sum to 10 (from index 0 to 3), then the subset of the right is 5 and the right sum is 5.

You will be given an array of integers and must determine whether there is a subset that meets the criteria.

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

It has the following:

- an array of integers

Input Format

The first line contains n , the number of test cases.
The next n lines of the test case contain the array arr .
For each test case, the number of elements in the array arr .
The next line contains the number of elements in the array arr .

Constraints

$1 \leq n \leq 10$
 $1 \leq m \leq 10^5$
 $1 \leq arr[i] \leq 10^5$
 $0 \leq i < n$

Output Format

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

Sample Input 0

2
3
1 2 3
4
1 2 3 4

Sample Output 0

YES
NO

Explanation 0

For the first test case, the subset $arr[0] + arr[1] + arr[2]$ has the same sum as the right side.
For the second test case, $arr[0] + arr[1] + arr[2]$ has the same sum as the right side.

Sample Input 1

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

Sample Output 1

YES
YES
YES

Explanation 1

In the first test case, $arr[0] + arr[1] + arr[2]$ has the same sum as the right side.
In the second test case, $arr[0] + arr[1] + arr[2]$ has the same sum as the right side.
In the third test case, $arr[0] + arr[1] + arr[2]$ has the same sum as the right side.

Answer: (currently empty 0%)

123456789101112131415161718192021222324252627282930313233343536373839404142434445464748495051525354555657585960616263646566676869707172737475767778798081828384858687888990919293949596979899100

```
1 // Merge Sort
2
3 // Merge Sort
4 int mergeSort(int arr[], int left, int right, int temp[])
5 {
6     if (left < right)
7     {
8         int mid = (left + right) / 2;
9         mergeSort(arr, left, mid, temp);
10        mergeSort(arr, mid + 1, right, temp);
11        merge(arr, left, mid, right, temp);
12    }
13    return arr;
14 }
15
16 // Merge Sort
17 void mergeSort(int arr[], int left, int right, int temp[])
18 {
19     if (left < right)
20     {
21         int mid = (left + right) / 2;
22         mergeSort(arr, left, mid, temp);
23         mergeSort(arr, mid + 1, right, temp);
24         merge(arr, left, mid, right, temp);
25     }
26 }
27
28 // Merge Sort
29 void mergeSort(int arr[], int left, int right, int temp[])
30 {
31     if (left < right)
32     {
33         int mid = (left + right) / 2;
34         mergeSort(arr, left, mid, temp);
35         mergeSort(arr, mid + 1, right, temp);
36         merge(arr, left, mid, right, temp);
37     }
38 }
```

Input	Expected	Got
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10
11	11	11
12	12	12
13	13	13
14	14	14
15	15	15
16	16	16
17	17	17
18	18	18
19	19	19
20	20	20
21	21	21
22	22	22
23	23	23
24	24	24
25	25	25
26	26	26
27	27	27
28	28	28
29	29	29
30	30	30
31	31	31
32	32	32
33	33	33
34	34	34
35	35	35
36	36	36
37	37	37
38	38	38
39	39	39
40	40	40
41	41	41
42	42	42
43	43	43
44	44	44
45	45	45
46	46	46
47	47	47
48	48	48
49	49	49
50	50	50
51	51	51
52	52	52
53	53	53
54	54	54
55	55	55
56	56	56
57	57	57
58	58	58
59	59	59
60	60	60
61	61	61
62	62	62
63	63	63
64	64	64
65	65	65
66	66	66
67	67	67
68	68	68
69	69	69
70	70	70
71	71	71
72	72	72
73	73	73
74	74	74
75	75	75
76	76	76
77	77	77
78	78	78
79	79	79
80	80	80
81	81	81
82	82	82
83	83	83
84	84	84
85	85	85
86	86	86
87	87	87
88	88	88
89	89	89
90	90	90
91	91	91
92	92	92
93	93	93
94	94	94
95	95	95
96	96	96
97	97	97
98	98	98
99	99	99
100	100	100